Application - Application Architecture

Principle 2.00.01 Architect applications to mimic business.

Rationale:

- The business process must be completely understood, requiring each business event be identified and the work unit servicing each event be identified.
- The organization must understand how work units cooperate to supply value when business events occur.
- The logical boundaries for applications should be drawn around units of work.
- Each unit of work is implemented with a collection of related business rules.
- Applications should respond to business events by invoking business rules.
- The state can take advantage of distributed systems capabilities to make application services available where the work takes place (i.e., in the geographic location where service must be provided).

Application - Application Architecture

Principle 2.00.02 Design applications to be highly granular and loosely coupled

Rationale:

- The designer should allow for the possibility of re-partitioning an application in the future.
- Being highly granular and loosely coupled provide flexibility in physical implementation (i.e., in the deployment of application components on different platforms).
- Highly granular, reusable application components are key to increased productivity and to rapid application deployment. The Componentware Architecture supports a highly granular design.
- The Application Communication Middleware Architecture supports a loosely coupled design.

Application - Application Architecture

Principle 2.00.03 Plan for extensibility and scalability.

Rationale:

- Most applications evolve to support new business requirements.
- Extensibility provides functional scalability.

Application - Application Architecture

Principle 2.00.04 Design application to reuse components.

Rationale:

- Applications should be built by assembling and integrating existing components, rather than by creating custom code. Shrinking cycle times do not allow for artisan programming. (See Chapter 5 - Componentware Architecture.)

- Managing component reuse is supported by the System Management Architecture. (See Chapter 11 - Systems Management Architecture.)

Application - Application Architecture

Principle 2.00.05 Select tools based on Application Architecture.

Rationale:

- There is no such thing as one application tool that satisfies all application requirements.
- Most tools, such as user interfaces, business rules, end user reporting, on-line analytical processing, and multimedia, are oriented toward different areas of development.
- Integrated tools may blur logical application boundaries.
- Organizations need a suite of development tools to solve a variety of problems.
- Historically, project teams selected tools (e.g., Visual Basic, PowerBuilder, SQLWindows) first, and then had to live with the architecture those tools supported. That led to the problem of the tools driving the architecture (and thus the business) rather than the business requirements mandating the tools.
- Tools are just now becoming available for end-to-end design, development, and deployment of N-tier applications. As they mature, they may become the best of breed for all application tiers.

Application - Application Architecture

Principle 2.00.06 Define specific roles for programmers.

Rationale:

Each of the following classes of programmers can specialize and use the tools best suited to their roles:
- Business rule programmers.
- User interface programmers.
- Data access programmers.

Application - Application Architecture

Principle 2.00.07 N-tier Service Oriented Architecture enables rapid application deployment.

Rationale:

Time savings, required for rapid deployment, can only occur if:
- The application is comprised of re-usable components.
- The application architecture is documented and well understood.
- The infrastructure to support the application is in place.

Application - Application Architecture

Principle 2.00.08 Document the Architecture.

Rationale:

- Application architecture mimics and supports business processes, and must change when business processes change. Changes occur more frequently in business processes than in the data required to support the business.
- Project teams must focus on the process that creates the data at least as much as they focus on the data itself.
- Application Architecture is as important as Data Architecture (see the Data Architecture chapter) because data is a result of business processes.

Application - Designing and Developing Applications

Standard 2.01.01 Isolate Customizations to Purchased Software.

Rationale:

- Isolate customizations into separate modules from the purchased software itself to improve the ability to upgrade and move to new releases as required over time. For purchased line-of-business applications, loosely couple custom developed modules from the standard application software.

Application - Designing and Developing Applications

Best Practice 2.01.01 Design for the N-tier service oriented architecture.

Rationale:

- While many of the problems inherent in the state's existing monolithic and two-tier applications can be overcome by implementing applications with a three-tier architecture, large, complex projects that are anticipated to have high usage volumes and/or long life spans will be better served by an N-tier service oriented architecture.
- N-tier applications are easily modified to support changes in business rules.
- N-tier applications are highly scaleable.
- An N-tier architecture offers the best performance of any application architecture.
- Any combination of user interfaces (e.g., character, graphical, web browser, and telephone interfaces) may be implemented in an N-tier application.
- N-tier applications are less expensive to build and maintain because much of the code is pre-built and shared by other applications (see Chapter 5 - Componentware Architecture).

Application - Designing and Developing Applications

Standard 2.01.02 Develop 3-tier or N-tier Applications.

Rationale:

- All new agency applications should be developed using 3-tier or N-tier architecture in order to maximize flexibility and scalability.
- The logical separation of the tiers for: user interface(s); business rules; and data access code allows for simple, straightforward additions to each of the three tiers without undue impacts on the others.
- The logical separation of the tiers also allows for changing the platforms where the tiers are deployed, resulting in a high degree of scalability. As transaction loads, response times, or throughputs change, a tier can be moved from the platform on which it executes

to another, more powerful platform - or be spread over multiple machines - without impacting the other tiers.
- While many of the problems inherent in the state's existing monolithic and two-tier applications can be overcome by implementing applications with a three-tier architecture, the goal should always be true, N-tier applications.
- Large, complex projects that have high usage volumes and/or long life spans will be better served by an N-tier service oriented architecture.
- The maximum benefits of an N-tier architecture are realized when many N-tier applications are deployed across the state, sharing common software services and offering multiple user interfaces.

Application - Designing and Developing Applications

Best Practice 2.01.02 Do not focus on platforms or deployment.

Rationale:

- Developers should not focus on where application components will execute (i.e., where they will be deployed) or what platforms they will execute on.
- System designers and operations support staff should make deployment decisions.
- Developers must avoid platform-specific or hard-coded interfaces that are difficult to change.

Application - Designing and Developing Applications

Best Practice 2.01.03 Generalize application interfaces.

Rationale:

- Generalize application interfaces.
- The code providing input and output to the user interface should be designed to provide input and output to as wide a range of interfaces as possible. This should include other applications as well as other types of user interfaces.
- Do not assume that application components will always be accessed via a graphical user interface (or any other user interface).
- Avoid assuming a specific page size, page format, layout language or user language whenever possible.

Application - Designing and Developing Applications

Standard 2.01.03 Avoid Common Gateway Interface (CGI) for business logic or to publish information to the Web.

Rationale:

- The Common Gateway Interface (CGI) does not scale, is not portable and is not easily integrated with application servers. Avoid use of CGI for information publishing, back-end applications or data access.
- Publishing information to the web with HTML or XML via java servlets reduces overhead and works in conjunction with EJB-based components.
- The use of ASP or other HTML publishing is acceptable for publishing only (not business logic) but JSP and Servlets are preferred.

Application - Designing and Developing Applications

Best Practice 2.01.04 Assign responsibility for business rules to business units.

Rationale:

- Assign responsibility for defining and maintaining the integrity of business rules to business units.
- IT staff is responsible for coding and administering the software that implements business rules in the network.
- The business units are responsible for the definition and integrity of business rules, and for communicating changes in business rules to IT.
- Every business rule should be assigned to a custodian.

Application - Designing and Developing Applications

Best Practice 2.01.05 Make business rules platform-neutral.

Rationale:

- Implement business rules in a non-proprietary, cross-platform language.
- This approach provides platform independence and portability.

Application - Designing and Developing Applications

Best Practice 2.01.06 Implement business rules as discrete components.

Rationale:

- Implement business rules as discrete executable components or services (see Chapter 5 - Componentware Architecture).

Application - Designing and Developing Applications

Best Practice 2.01.07 Access data through business rules.

Rationale:

- Each agency owns its own data and controls the access to it. By designing applications so business rules permit access to data, the agency will maintain better control over access.
- Data is created and used by business processes. In computer applications, data must be created, used by, and managed by the application component that automates the business process.
- Accessing data in any way other than by business processes bypasses the rules of the module that controls the data. Data is not manages consistently if multiple processes or users access it.
- Federated data should be used wherever possible to assure data accuracy and simplify data management.

Application - Designing and Developing Applications

Best Practice 2.01.08 Achieve working system first.

Rationale:

- Once the detailed application design is complete, concentrate on achieving a working system utilizing off-the-shelf components whenever possible. This will allow the system to be tested first and then optimized later.

Application - Designing and Developing Applications

Best Practice 2.01.09 Design for manageability.

Rationale:

- Design applications so they can be managed using the enterprise's system management practices and tools (see the Systems Management chapter).
Applications and their components require the following management functions:
- Software distribution.
- Start-up, shutdown, and restart of components.
- Starting multiple instances of a component.
- Configuration of components.
- Logging of component operations.
- Communication of errors, exceptions, and unexpected events.
- Security.
- Installation, removal, and update of application modules.
- Version control.

Application - Designing and Developing Applications

Best Practice 2.01.10 Adopt coding standards.

Rationale:

Adopt coding standards, in all languages, on all platforms.
Coding standards make debugging and maintenance easier. They should address (but not be limited to):
- Naming conventions for variables, constants, data types, procedures and functions.
- Code flow and indentation.
- Error and exception detection and handling.
- Source code organization, including the use of libraries and include files.
- Source code documentation and comments.
- Even the earliest code developed in a project should adhere to the standards.

Application - Designing and Developing Applications

Best Practice 2.01.11 Design for ease of testing.

Rationale:

- Design application components so they can be easily tested and debugged.
- Testing is a critical step in the development of client/server applications.
- Application components with consistent interfaces are easier to test on an application-wide basis.
- Error-handling, tracing, and checkpointing should be included.

- These functions should be implemented in the earliest phases of development.

Application - Managing Applications

Standard 2.02.01 ITS has adopted an SNMP-compliant NSM tool. All applications deployed must be designed to be managed by SNMP. (Note: any deviations must be explicitly approved by the IRMC.)

Rationale:

- By standardizing on SNMP as the instrumentation protocol, there is an opportunity for the state to benefit from reusing management instrumentation code.

Application - Managing Applications

Best Practice 2.02.01 Application management code is a candidate for re-use.

Rationale:

- Application management code is a good candidate for re-use because it should be standardized within and across applications.
- Spend time designing code to support application management requirements. This code should be reused, rather than reinvented, in every application.
- Application management is easiest if application development teams are provided code templates that include most of the basic application management functions.
- Once the statewide NSM tool is in place, the application templates should include code to enable applications to be managed by the NSM tool.

Application - Managing Applications

Best Practice 2.02.02 Design for end-to-end management.

Rationale:

- Manage the application as a whole entity by managing every component of the application and everything each component depends on. Application developers must instrument every component of the application to facilitate its management.
- Application dependencies include infrastructure (e.g., middleware, databases, and networks), other applications, and shared software components. Application teams must specify these dependencies when an application is deployed.
- Application reporting should be standards based and must be compatible with the state's NSM tool.

Application - Managing Applications

Best Practice 2.02.03 Design for proactive - rather than reactive - application management.

Rationale:

- Design for proactive -- rather than reactive -- application management.
- Proactive application management supports the business better. With proactive management, applications report potential problem conditions at predefined thresholds,

before errors occur. This gives system administrators the opportunity to take corrective action to prevent an application from failing.
- While applications can be managed by administrators responding to errors, the ideal management is automatically undertaken by the NSM tools, and is proactive.
- Use thresholds to provide early alert to possible error conditions. For example, rather than sending an alarm when an application fails because its database table is full, send an alarm when the table is 90% full, so corrective action can be taken to prevent a business-impacting outage.
- Reactive application management is better than no application management at all. Reactive management is when administrators respond to errors and outages reported by applications after they have occurred.

Application - Managing Applications

Best Practice 2.02.04 Instrument applications to report the information necessary to manage them.

Rationale:

- Applications should report status, performance statistics, errors, and conditions. Decide at design time what status events the application should report to users (e.g., erroneous input); to application managers (e.g., database table 90% full); and to both (e.g., can't find needed file).
- Operations staff must be provided procedures for dealing with all conditions that are detected and reported. For example, if an application reports it can no longer access its database, operations staff must have instructions for handling the situation.
- At design time, decide the specific reporting requirements of an application module. Different applications may have different management needs, depending on their respective impact on the business the applications support.
- Applications should only report. Interpreting the reports and deciding on the appropriate response should be performed external to the application, by agents and the NSM framework.
- Application reporting should include run-time tracing to assist trouble-shooting operational problems. Tracing should be able to be turned on and off by administrators.
- If no NSM environment exists, applications should still report status to local log files that can be monitored by administrators. Applications should still be able to read and respond to commands from administrators.

Application - Managing Applications

Best Practice 2.02.05 Instrument applications to facilitate administration.

Rationale:

- Instrument applications to receive and process commands from administrators.
- Decide at design time what control operators and NSM tools should have over application components.
- Design applications to read and respond to commands from system administrators. Commands may include, but are not limited to, shut down, shutdown and restart, reconfigure yourself, and turn tracing on or off.

- Make application configurations parameter-driven, so applications can be reconfigured without recompiling and redistributing code.

Application - Accessibility

Standard 2.03.01 Utilize the latest version of the World Wide Web Consortium Web Content Accessibility Guidelines.

Rationale:

- The W3C is the international standards body for such protocols as HTML, XML, and CSS. The accessibility guidelines mirror Federal and International requirements for accessibility. The URL for the document can be found at http://www.w3.org/TR/WAI-WEBCONTENT.
- New web sites must utilize this standard 6 months after adoption.
- Existing Web sites must meet the standard 18 months after adoption.

A note about the checkpoints contained in the standard:

- Each checkpoint has a priority level assigned by the W3C Working Group based on the checkpoint's impact on accessibility.

[Priority 1]
- A Web content developer must satisfy this checkpoint. Otherwise, one or more groups will find it impossible to access information in the document. Satisfying this checkpoint is a basic requirement for some groups to be able to use Web documents.

[Priority 2]
- A Web content developer should satisfy this checkpoint. Otherwise, one or more groups will find it difficult to access information in the document. Satisfying this checkpoint will remove significant barriers to accessing Web documents.

[Priority 3]
- A Web content developer may address this checkpoint. Otherwise, one or more groups will find it somewhat difficult to access information in the document. Satisfying this checkpoint will improve access to Web documents.

Some checkpoints specify a priority level that may change under certain (indicated) conditions

Application - Accessibility

Best Practice 2.03.01 Ensure graceful transformation.

Rationale:

- Pages that transform gracefully remain accessible despite any of the constraints described in the introduction, including physical, sensory, and cognitive disabilities, work constraints, and technological barriers.
- Separate structure from presentation.
- Provide text (including text equivalents). Text can be rendered in ways that are available to almost all browsing devices and accessible to almost all users.

- Create documents that work even if the user cannot see and/or hear. Provide information that serves the same purpose or function as audio or video in ways suited to alternate sensory channels as well.
- This does not mean creating a prerecorded audio version of an entire site to make it accessible to users who are blind. Users who are blind can use screen reader technology to render all text information in a page.

Application - Accessibility

Principle 2.03.01 Emerging technologies that improve accessibility should be utilized to the maximum extent possible.

Rationale:

- Technology is evolving rapidly. New access problems and solutions are appearing on a nearly daily basis.
- The rapid development of technology makes any determination of current technical feasibility extremely short lived.
- Many standards contained in this architecture are based upon human need, rather than specific solutions.
- As technology moves forward, solutions that improve accessibility are always best practices.

Application - Accessibility

Best Practice 2.03.02 Create documents that do not rely on one type of hardware.

Rationale:

- Pages should be usable by people without utilizing a mouse, with small screens, low resolution screens, black and white screens, no screens, with only voice or text output, etc.

Application - Accessibility

Principle 2.03.02 Accessibility should be proportional to the existing and potential needs.

Rationale:

This principle is intended to assure that requirements do not create unforeseen inefficiencies and create cost without significant benefit.
- Systems should be assessed for their need to be accessible to achieve the highest value.
- No standard is applied so as to go beyond what is necessary to achieve its objective.
Undue Burden means significant difficulty or expense. In determining whether an action would impose an undue burden on the operation of the agency in question, factors to be considered include:
- The nature and cost of the action needed to make a system accessible;
- The overall size of the agency's program and resources, including the number of employees, number and type of facilities, and the size of the agency's budget;
- The type of the agency's operation, including the composition and structure of the agency's work force;

- The impact of such action upon the resources and operation of the agency.
- Statewide or Citizen services should always consider accessibility.

Application - Accessibility

Principle 2.03.03 Compatibility increases accessibility.

Rationale:

- Systems that maximize the accessibility functions of all tiers provide information in its most accessible form.
- Many people use "accessors" with limited capability to process information.
- Compatible multi-tier platforms (i.e. Microsoft's Windows Server/Professional/CE or Linux/Palm) provide opportunities to maximize accessibility.
- Software and hardware should be compatible with existing standards in Adaptive Technology.
- Government, or industry standards for hardware and software interfaces are positive, but not definitive, indicators that E&IT and accessibility technologies that are being considered for use will be compatible.

Application - Accessibility

Principle 2.03.04 Separate the Presentation Layer from the Content Layer.

Rationale:

- Accessible systems allow the client to process content in the form that best suits the client and user's capabilities.
- Users of a system are trying to access INFORMATION, not PRESENTATION.
- Systems that are utilized specifically for presentation, such as public relations material, should have their content available in an accessible format that is fundamentally as easy to obtain and use as the manner in which the content is provided inaccessibly.
- Documents should not be stored and cataloged in a manner that mandates a particular presentation (e.g. Adobe PDF), or is not in itself accessible (e.g. PCDocs).

Application - ComponentWare

Principle 2.04.01 Componentware Architecture facilitates the reuse of components across the enterprise.

Rationale:

- Reusable components increase the productivity of the application development departments within the enterprise.
- Sharing components across the enterprise greatly increases the ability of the system to meet the changing needs of the business.
- The use of proven components enhances the accuracy of information processing.

Application - ComponentWare

Principle 2.04.02 The focus of Componentware Architecture is to improve business performance.

Rationale:

- A component-based development strategy enables adaptive systems to meet the changing business needs and technical environments.
- A component-based development strategy aligns information technology with the commonly used functions of the business.

Application - ComponentWare

Principle 2.04.03 Shareable components must be callable by any application.

Rationale:

- Reusing existing shared components eliminates duplication of development, testing, and maintenance effort.
- Reusing existing shared components eliminates processing inconsistencies because business rules are maintained in one piece of code.
- Use of components reduces the time and effort required for developing and updating applications.

Application - ComponentWare

Principle 2.04.04 Components should have ownership and maintenance responsibilities assigned.

Rationale:

- The responsibility for the development and maintenance of the component remains with the application development team.
- The responsibility for the definition of a component should is within the business organization within the enterprise that is responsible for the function.

Application - ComponentWare

Principle 2.04.05 A component should implement a single business rule or function, or a small set of related business rules or functions.

Rationale:

- To maximize component reusability each should contain a single function.
- Each component must have a single point of entry and a single point of exit.

Application - ComponentWare

Principle 2.04.06 Develop reusable testing suites for every component. A component testing suite contains special programs needed for calling a component as well as sample input and example output data for verifying results.

Rationale:

- Testing suites would be developed that can be reused any time a particular component is modified.
- The testing suites will be maintained and managed the same as any other component.

Application - ComponentWare

Principle 2.04.07 New components must be platform independent.

Rationale:

- Components must be developed so they can be deployed on any supported platform.
- If the business needs change or a new platform is required, the component should easily migrate to a new platform.

Application - ComponentWare

Principle 2.04.08 Purchase rather than build components whenever possible.

Rationale:

- Purchased components must be capable of being implemented in a service-oriented environment, (i.e., can be integrated into an N-tier environment with a published Application Programming Interface (API)).
- Components should be purchased whenever possible, such as class libraries, allowing developers to focus on the development of specialized business rule components.

Application - ComponentWare

Principle 2.04.09 Design components to be fully self-contained.

Rationale:

- All necessary validation, error detection and reporting capabilities, logging/debugging/tracing functionality, monitoring and alert functionality, and systems management capabilities must be incorporated in the component.
- This facilitates operation, administration, and maintenance functions.

Application - ComponentWare

Principle 2.04.10 Establish guidelines to optimize performance.

Rationale:

- Guidelines for request and reply message lengths should be established to avoid undue network traffic and performance problems.
- Components should be compiled into a binary executable format, not interpreted.

Application - Component Reuse

Standard 2.05.01 No vendor proprietary API calls for infrastructure security services. Use Generic Security Services-API (GSS-API) or CDSA compliant API calls and products.

Rationale:

- Applications requiring security services prior to CDSA products or services being available can use the GSS-API.
- The GSS-API is an IETF standard (RFC 2078) and supports a range of security services such as authentication, integrity, and confidentiality.

- It allows for plug-ability of different security mechanisms without changing the application layer.
- It is transport independent, which means it can be used with any underlying network protocol.
- Applications using GSS-API can be retrofit to a CDSA foundation without major modifications, therefore providing an interim step to CDSA based services.

Application - Component Reuse

Best Practice 2.05.01 Establish a reuse methodology for the identification and implementation of components.

Rationale:

A methodology that supports reuse contains the following steps:
- Classify the business requirements by service type (e.g., application, interface, support, or core).
- Search the repository for reusable components that support the business or functional requirements.
- Analyze candidate components to ensure they fully meet the requirements.
- Incorporate the selected components into the new or re-engineered application using standard API's.
- Harvest new components from new or existing applications that have not been componentized yet. Placing the new component information into the repository.
- Incorporate the reuse methodology into the system development life cycle.

To successfully implement the Componentware Architecture, the Network and Middleware Architecture must be in place. For more information, refer to the Network Architecture and Middleware Architecture chapters.

Application - Component Reuse

Best Practice 2.05.02 Establish a component review board to identify common components.

Rationale:

- Components used by multiple business units must be commonly understood and consistently referenced by all business users.
- Component development can be achieved through the context of projects.
- The review board should start with small, achievable, and strategic projects.
- In order to create reusable components, cooperation is needed among the business process owners.
- A framework needs to be put in place that allows for:
- Centralized management of reusable, shareable components.
- Design reviews of new and existing projects for reusable components.
- Enterprise access to information about reusable components.

Application - Component Reuse

Best Practice 2.05.03 Establish component design reviews of all ongoing projects.

Rationale:

- Determine whether the business requirements are met by any existing components.
- If the business requirements are not met by existing components, determine if there are any components that could be expanded to satisfy the business requirement (without compromising the reusability of a component).
- Analyze existing applications for potential additions to the component repository.
- Access should be provided to the proper members of application development projects to reference the component repository in order to actively research application requirements.

Application - Component Reuse

Best Practice 2.05.04 Establish a repository for maintaining the information about available reusable components.

Rationale:

- The repository provides a place to store documentation about the component API's.
- The repository should be made available to all application developers as a tool for performing their jobs.

Application - Component Reuse

Best Practice 2.05.05 Components should have ownership and maintenance responsibilities assigned.

Rationale:

- The responsibility for the development and maintenance of the component remains with the application development team.
- The responsibility for the definition of a component is within the business organization within the enterprise that is responsible for the function.

Application - Component Reuse

Best Practice 2.05.06 Every component must have a published API.

Rationale:

- A published API defines the public interface for a component or service. The API is how other applications will communicate with the component. Documentation should include input and output parameters, which parameters are required, which parameters are optional, and the lengths and types of the parameters.
- The API should be entered into the component repository that is available to all application developers.

Application - Component Reuse

Best Practice 2.05.07 Harvest components from existing applications to initially build the component repository.

Rationale:

- Legacy applications are a good resource for building a component repository.
- There is no need to reinvent a process or piece of functionality if software already exists that performs the desired function.
- If feasible, develop a wrapper that defines the API for the service and allows the legacy application to become a reusable component.

Application - Component Reuse

Best Practice 2.05.08 A component should implement a single business rule or function, or a small set of related business rules or functions.

Rationale:

- To Maximize component reusability each should contain a single function.
- Each component must have a single point of entry and a single point of exit.

Application - Component Reuse

Best Practice 2.05.09 Develop reusable testing suites for every component. A component testing suite contains special programs needed for calling a component as well as sample input and example output for verifying results.

Rationale:

- Testing suites would be developed that can be reused any time a particular component is modified.
- The testing suites will be maintained and managed the same as any other component.

Application - Component Reuse

Best Practice 2.05.10 Adopt effective component management methodologies, including the tools to support component reuse.

Rationale:

- In a distributed development environment, there must be a methodology in place for managing the available components.
- It must include the steps necessary for identifying, defining, and developing reusable components.
- If this methodology is not in place then reuse will be very difficult to implement.

Application - Component Services

Best Practice 2.06.01 Component services should be callable by any application or any other component.

Rationale:

- Components must be designed and developed with the understanding that the process that invokes it may or may not be developed in the same language or in the same environment.

- A component should be callable from any supported language on any supported platform.

Application - Component Services

Standard 2.06.01 Custom developed application components must be written in a portable, platform-independent language, such as C, C++, or Java.

Rationale:

- Application components written in a portable language are easier to move from one platform to another because they require fewer modifications to conform to the new host. This portability allows an application to be more adaptive to changes in platform technology.

Application - Component Services

Best Practice 2.06.02 New components must be platform independent.

Rationale:

- Components must be developed so they can be deployed on any supported platform.
- If the business needs change or a new platform is required, the component should easily migrate to a new platform.

Application - Component Services

Standard 2.06.02 Use statewide security infrastructure when available.

Rationale:

- Security services will be provided as a statewide infrastructure service.

Application - Component Services

Best Practice 2.06.03 Purchase rather than build components whenever possible.

Rationale:

- Purchased components must be capable of being implemented in a service-oriented environment, (i.e., can be integrated into an N-tier environment with a published Application Programming Interface (API)).
- Components shjuold be purchased whenever possible, such as class libraries, allowing developers to focus on the development of specialized business rule components.

Application - Component Services

Standard 2.06.03 Statewide infrastructure services must be Common Data Security Architecture version 2.0 (CDSA v2.0) compliant.

Rationale:

- The CDSA version 2.0 architecture is an Open Group specification for providing security services in a layered architecture and managed by a Common Security Services Manager (CSSM). CDSA provides a management framework necessary for integrating

security implementations. Version 2.0 of the specification is a cross-platform architecture, which includes a testing suite for inter-operability testing.
- A wide range of vendors has announced support for the specification and products for a broad set of platforms can be expected.
- Security protocols such as SSL, S/MIME, IPSec can all be built from a CDSA base.

Application - Component Services

Best Practice 2.06.04 Design components to be fully self-contained.

Rationale:

- All necessary validation, error detection and reporting capabilities, logging/debugging/tracing functionaility, monitoring and alert functionality, and system management capabilities must be incorporated in the component.
- This facilitates operation, administration, and maintenance functions.

Application - Component Services

Best Practice 2.06.05 Establish guidelines to optimize performance.

Rationale:

- Guidelines for request and reply message lengths should be established to avoid undue traffic and performance problems.
- Components should be compiled into a binary executable format, not interpreted.

Network - Network Architecture

Principle 3.00.01 The network provides a communications infrastructure for distributed computing.

Rationale:

- The world is increasingly connected. A network environment provides access to a wide spectrum of information, applications, and resources.
- Any product or application not architected for a networked environment is limited long-term.
- The network provides the delivery mechanism for distributed services in an n-tier architecture.

Network - Network Architecture

Principle 3.00.02 A single integrated wide area network (WAN) is the backbone of an enterprise architecture and supports a variety of communication requirements including voice, data, image, and video.

Rationale:

- It allows access to a wide spectrum of information, application and system resources regardless of location or business unit. Thus, access to resources can be obtained in a timely and efficient manner by appropriate requesters when and where needed throughout the enterprise.

- It expands the scope of an organization domain by allowing them to reach out to customers and suppliers through access to the Internet and through the provision of dial-in/dial-out services.
- It acts as the delivery mechanism for the distributed computing services required by the fast-paced, dynamic business.

Network - Network Architecture

Principle 3.00.03 Networks should be available seven days a week and twenty-four hours a day.

Rationale:

- Networks provide an increasingly important and necessary role in the execution of business functions and processes. The availability of the network seven days a week and twenty-four hours a day must be maintained in a consistent and complete manner.
- Networks consist of and rely on many interrelated and often highly complex components distributed across a wide geographic area. Failure of any single component can have severe adverse effects on one or more business applications or services.
- Reliable networks contain no single point of failure. Networks are comprised of many components, and are only as reliable as the weakest link. Reliability must be built-in, not added-on.
- Bandwidth must be sufficient to accommodate new and expanding applications, different types of data (e.g., voice, data, image, and video), and a variety of concurrent users.
- The network must support software distribution and installation to a widely dispersed user community.
- The network must be designed to minimize latency. Data must pass across the network in a timely manner so that business decisions can be based on up-to-date information.

Network - Network Architecture

Principle 3.00.04 A statewide network must be based on common, open, vendor-neutral protocols.

Rationale:

- An open, vendor-neutral protocol provides the flexibility and consistency that allows agencies to respond more quickly to changing business requirements.
- An open, vendor-neutral network allows the state to choose from a variety of sources and select the most economical network solution without impacting applications.
- This approach supports economic and implementation flexibility because technology components can be purchased from many vendors. This insulates the state from unexpected changes in vendor strategies and capabilities.
- Applications should be designed to be transport-independent.

Network - Network Architecture

Principle 3.00.05 User access should be a function of authentication and authorization, not of location.

Rationale:

- All users must obtain authentication via a user identification method consistent with the standards and usage guidelines set by the enterprise.
- Authorization of users must be performed according to the security rules of the enterprise and the local business unit.
- In order to perform their job functions, users need to access services available from multiple sites within the enterprise, from a variety of public and private networks, and from the Internet.

Network - Local Area Network

Standard 3.01.01 The standard for LAN cabling is Category 5, 6, or 7 Unshielded Twisted Pair (Cat 5 UTP, Cat 6 UTP, or Cat 7 UTP).

Rationale:

- CAT 5/6/7 UTP can be certified to carry 10/100/1000 MBPS of data.
- It is a industry standard wiring plan and has the support of the IEEE.
- Wiring, cable, connector, and equipment vendors have standardized on this cabling.

Network - Local Area Network

Best Practice 3.01.01 Networks must be positioned for future growth in traffic and expansion of services such as voice and video.

Rationale:

- The increasing investment of funds in network infrastructures dictates that the life span of each additional component or enhancement be as long as possible. This can be accomplished if the design supports current needs but includes an anticipated growth potential. For example, installing Category 5 cabling today to run a 10mbps network positions a site to upgrade to a 100mbps speed in the future without replacing the cabling.
- As businesses expand, networks expand. A flexible, open network design will allow a business to minimize the costs and disruptions of configuration management while providing timely and responsive network changes when and where required.

Network - Local Area Network

Standard 3.01.02 The standard for standard link layer access protocol is Ethernet, IEEE 802.3 Carrier Sense Multiple Access/Collision Detection Access Method (CSMA/CD).

Rationale:

- Widely accepted format.
- Reliable, the protocol has been used for years and is very stable.
- Scaleable, faster versions are currently emerging to help manage the increase of data flow.
- 1000BaseT Gigabit Ethernet has the bandwidth necessary to support the needs of future voice and video requirements.

Network - Wide Area Network

Standard 3.02.01 The standard protocol technology is TCP/IP.

Rationale:

- Open protocol.
- Allows Internet access.
- Allows creation of Intranets and VPNs.

Network - Network-Centric Applications

Best Practice 3.03.01 Include network expertise on the requirements and design teams.

Rationale:

- Including network expertise ensures correct planning, documentation, and standard practices are followed.
- Requirements definition should include application performance, as well as capacity planning for network usage (based on the predicted number and size of transactions).
- Define any special networking requirements or constraints and perform the associated network design before development tools are selected. Otherwise, the tools used may not support the network architecture required to support the business.
- The network can be modified (upgraded) while applications are under development.
- Performance and the cost to move information should be balanced during application design. Multiple perspectives of a cross-functional group can ensure all viable options are considered.

Network - Network-Centric Applications

Best Practice 3.03.02 Design network-neutral applications.

Rationale:

- Isolate the application code from the network specific code so business rules and data access code can be redeployed on a different platform, if necessary.
- Code to a middleware API, not to the network API.
- For a network to remain scaleable and portable, applications must be developed without regard to the type of network (i.e. WAN or LAN) they are to be deployed on.
- Network-specific design (e.g., wireless or guaranteed high-bandwidth) should only be performed when business requirements dictate.

Network - Network-Centric Applications

Best Practice 3.03.03 Minimize data movement.

Rationale:

- When possible, schedule heavy network use for off-peak hours. For example, where requirements for data freshness permit, perform database synchronization at night.
- Data warehouses typically are used for decision support applications requiring large amounts of data to be transferred through the network.

- When replicating databases, consider partitioning and distributing subsets, rather than duplicating the entire master database.
- Decoupling the application layers provides the most efficient use of network resources by allowing the data access layer to be placed near the data.

Network - Network-Centric Applications

Best Practice 3.03.04 Consider the impact of middleware on network utilization.

Rationale:

- Perform all transaction commits locally, between the resource manager and the queue. Asynchronous store and forward messaging can limit the scope of a transaction. (See Figure 3-7.)
- Decouple transactions as allowed by business rules. Reconcile data at low-cost times.
- Using store and forward, work can occur at a site even if the network link is down.
Long binary or text value

Network - Network-Centric Applications

Best Practice 3.03.05 When data has to be distributed to multiple points (e.g., software and content distribution), move it once and only once across each data link.

Rationale:

- Use push technology, rather than using client polling. It overloads servers and network links to servers.
- Use multicast, rather than broadcast, to distribute messages to multiple points.

Network - Network-Centric Applications

Best Practice 3.03.07 Perform performance measurement and load testing on distributed applications before deployment.

Rationale:

- Measure application performance often, especially before and after any component is moved to a different platform. This helps quantify the performance impact of the redeployment, and helps isolate any problems associated with a network link or platform.
- Use load testing tools that simulate many users accessing the application. This testing method will provide information that will not surface during single user test scenarios.
- Load testing will identify network bottlenecks (and application bottlenecks) before the application is deployed in the production environment.

Network - Network-Centric Applications

Best Practice 3.03.08 Deploy heavily used data sources "close" to the applications using them.

Rationale:

- "Close" does not imply physical proximity. It means deployed on platforms that have high-bandwidth connections between them. Do not perform heavy data movement across the WAN during peak hours.
- One of the biggest cost factors in designing a network is the transmission of the data over the communications system.
- For applications requiring very large amounts of data movement, try scheduling the execution of these queries to run during off peak hours to minimize the impact on network performance.

Network - Directory Services

Standard 3.05.01 Use the statewide directory services infrastructure.

Rationale:

Using the statewide directory services has several benefits:
- The infrastructure is simplified by providing a common interface to view and manage all available resources.
- Directory services are a critical component to statewide initiatives like E-mail and Electronic Commerce. The current enterprise directory is fault tolerant and highly available from any location that participates. Time,
distance, and location do not restrict access to the information contained within the services.
- Coordinated directory services will improve communication between our applications, databases, and network operating systems by providing consistent, reliable information in an efficient and effective manner.

Network - Directory Services

Best Practice 3.05.01 Implement a fault tolerant solution to provide 24-hour, 7-day availability to the enterprise directory.

Rationale:

- If the directory becomes inaccessible, the resources to which a user has rights become unavailable. Therefore, a directory mus be available at all times to accept authentication requests. This can be accomplished with a planned fail-over strategy to ensure that, if one server fails, another backup server can pick up the requests. This should include a replication strategy with hardware solutions that include disk or system duplexing, disk or system mirroring, disk arrays, and UPSs.

Network - Directory Services

Standard 3.05.02 Use the statewide directory services (NDS) for in-house developed applications to authenticate users.

Rationale:

- Novell NDS has become the State’s de Facto standard for enterprise directory services. This has been a
successful, well-coordinated effort that has been recognized internationally. NDS is available on several

platforms such as Microsoft NT, Sun Solaris, and many others. NDS also supports numerous access protocols such as LDAP, ODBC, Java, and ActiveX. Implementing NDS on the available platforms, using the supported access protocols, will achieve interoperability between these platforms and applications. It will also provide a single point of administration and authentication. Participation in the Statewide NDS tree is required for those NDS installations within the NCIIN whether from NDS on NetWare or NDS on the other available platforms. Where NDS is not currently available, such as in the mainframe environment, use the Service Broker security services for authentication.
- In an enterprise environment, issues such as server naming conventions, net ids, tree structures, etc. must be carefully coordinated and adhered to. These have been addressed in documents under F. Resources -
Distributed Computing Standards and Guidelines in this document.

Network - Directory Services

Best Practice 3.05.02 Purchased applications and operating systems should be directoryenabled.

Rationale:

- Securing applications and their operating environments is a significant challenge. Security is a natural
environment for the use of a directory. Applications can authenticate users to an external source by being
directory enabled. The directory is better suited to provide information on the level of security necessary.
Applications can be further enhanced when they are enabled to obtain an expanded set of information from the directory as appropriate. Thus making applications more modular and consolidating administration to a central location. For example, an application can gather employee information from the user object in the directory. This facilitates user authentication and authorization by making the resources on that platform available to the enterprise, when the appropriate rights are in place.

Network - Directory Services

Standard 3.05.03 Integrate homogeneous directories into a single tree.

Rationale:

- It is more efficient to link “like” directories into a single tree. Most vendors of directories have implemented
either the standards that currently exist or standards that have been proposed. These standards include a
mechanism to connect their directories together to build a single tree. This provides the optimum integration of Public Agency resources and people without regard to location. A single tree minimizes infrastructure costs while maximizing the potential for agencies to choose how they share resources with other agencies including local governments. The single tree approach also allows for improved fault tolerance and better performance especially for agencies with geographically dispersed operations. Joining a tree, regardless of manufacturer, must be coordinated with Information Technology Services.

- It is necessary to tie these single trees from various manufacturers to the authoritative enterprise directory in
order to provide authentication services to the authoritative enterprise directory. However, achieving
connectivity from one manufacturer's directory to another is complex and difficult. For example, tying one
Netscape directory to Novell's NDS can be done but is difficult to implement and maintain. The state currently has dozens of Netscape directories in place. The process would then need to be performed for each of them. However, tying all Netscape directories together into a single tree is fairly straightforward and facilitated through their product. Then the one Netscape tree can be tied to the one NDS tree. It is a complicated task but it is performed once. Through the Enterprise Directory Services Initiative, this interoperability between dissimilar directories will be implemented. This will be accomplished through the use of meta-directory technologies.

Network - Directory Services

Standard 3.05.04 Use the North Carolina Service Broker (NCSB) services for directory functions.

Rationale:

- As in-house applications are developed, we must make use of the services that are already available rather than to constantly build new ones. An enterprise directory services infrastructure provides an addressable security service for authentication and authorization as well as a repository for digital certificates.
- These services are addressable directly from the enterprise directory or through a service via the North Carolina Service Broker. For more information about the North Carolina Service Broker, refer to the Componentware Architecture chapter.

Network - Directory Services

Standard 3.05.05 Use the Federated Metadata Repository directory schema attributes and object classes.

Rationale:

- A directory is basically a database that has been tuned to perform massive reads and infrequent writes. Like other databases in our enterprise, directories and their elements must be federated. For example, where a person object class may have an attribute of "Pager Number", "Pager Number" should be registered in the Federated Metadata Repository and populated according to that definition. Therefore, when the directory is queried for that information, the data returned will be as expected. In the past there has been a tendency to populate currently unused directory attributes with data that is not consistent with that attribute. For example, there may be a requirement to enter a pager number in the directory for a user. If there is no attribute for "Pager Number", there may be a tendency to select an attribute that is unused such as "Title". Instead, extend the schema to include a new attribute that precisely defines the data that will be placed there and register it with the Federated Metadata Repository. Do not store inconsistent information in an unused attribute.

- For more information about the Federated Metadata Repository, refer to the Data Architecture chapter.

Network - Directory Services

Standard 3.05.06 Use the State Novell NDS enterprise directory as the authoritative source for directory information.

Rationale:

- An enterprise directory is not a single directory product. It will be made up of several directories from several
vendors. However, in an enterprise directory strategy one directory must be identified as the main directory and the authoritative source for all directory information. All other directories and applications in the enterprise look to it for complete re liable information.

Network - Directory Services

Standard 3.05.07 Populate directory objects according to the minimum attributes defined in Distributed Computing Standards and Guidelines.

Rationale:

- Any data source is only as good as the data it contains. If that data is missing, incorrect, or incomplete, the data source cannot be depended upon as an authoritative source for that type of information. A directory is no different. Directories have become much more than an authentication point for network users. In order to supply information on our users, network devices, and organizations, directories must be built in as complete and reliable manner as possible.
- The object attributes for Novell NDS are listed and explained in the Distributed Computing Standards and
Guidelines in the Novell Directory Services section at
http://www.state.nc.us/SIPS/services/help/documents/netware/novell.htm. The attributes are identified as Mandatory, Recommended, Optional, Unused, Automatic, Multi Valued, and Template.

Network - Directory Services

Standard 3.05.08 Use Lightweight Directory Access Protocol version 3 (LDAPv3) for directory access where strong security is not required.

Rationale:

- LDAPv3 is the industry standard lightweight access protocol and does not offer strong authentication or access controls. However, LDAPv3 can provide standards based access to directories for lookups, as a communication mechanism for synchronization tools, public key retrieval, and others. Commercial off-the-shelf (COTS) applications often require their own directories. Access to the application directory from outside or for the application to communicate with an external directory will require a standards based approach. Therefore, when purchasing CO TS applications, LDAPv3 compatibility is required. LDAPv3 also provides a standards based access to the directory for lookups, as a communication mechanism for synchronization tools, public key retrieval, and others.

Data - Data Architecture

Principle 4.00.01 Value and protect the state's data as one of the most critical assets in the organization.

Rationale:

- The successful delivery of government services depends on conclusions derived from accurate, well-maintained, and secure data.
- Protecting the value of the state's data is everyone's responsibility, especially business users and Information Technology staff.
- Leverage and maximize the use of data throughout the state.
- Implementing and enforcing data security is crucial to the organization.

Data - Data Architecture

Principle 4.00.02 Design an adaptive data infrastructure.

Rationale:

- Design the data infrastructure to easily accommodate changes in the data model and database technology. The data infrastructure is a crucial component of establishing an overall adaptive architecture.
- An adaptive data infrastructure provides extensibility in adding new functionality and facilitates vendor independence.

Data - Data Architecture

Principle 4.00.03 Design the enterprise Data Architecture so it increases and facilitates the sharing of data across the enterprise.

Rationale:

- Sharing of data greatly reduces data entry and maintenance efforts.
- Data sharing requires an established infrastructure for widespread data access. This includes integration with the Application, Componentware, Integration, Messaging, Network, and Platform Architectures.
- Consistent shared data definitions ensure data accuracy, integrity, and consistency.
- Data sharing reduces the overall resources required to maintain data across the enterprise.
- For more information, refer to the Federated Metadata topic in this chapter.

Data - Data Architecture

Principle 4.00.04 Design the enterprise Data Architecture so it is business driven, as opposed to technology driven, and aligned with the Application Architecture.

Rationale:

- By designing a Data and Application Architecture that is business driven as opposed to technology driven, an infrastructure is created that is adaptive and flexible to changes in business need.

- Data Architecture supports established business and data manageability requirements.
- Consider both business and application performance drivers.

Data - Data Architecture

Principle 4.00.05 Create provisions for supporting data in a distributed environment.

Rationale:

- In order to maintain a high level of data integrity and data quality, the Data Architecture must simplify the management of distributed data.
- A centralized repository of database-related information, (a.k.a. metadata), facilitates distributed data management.

Data - Data Architecture

Principle 4.00.06 Create and maintain roles and responsibilities within the distributed enterprise Data Architecture to facilitate the management of data. This requires a working relationship between the business user organizations and information services (IS).

Rationale:

The business users are responsible for establishing and maintaining the accuracy of data collected and entered into applications because these systems support the business need. Business responsibilities are to:
- Provide accurate business definitions of data.
- Develop enterprise-wide business views of shared data.
- Provide business drivers to support centralized data administration.
- Make federated metadata available.
- Define security requirements for data.
- IS Responsibilities are to provide a robust technical infrastructure that includes:
- Open, accessible, and adaptable database management systems (DBMSs).
- Centralized data administration.
- Data replication facilities.
- Backup and recovery.
- Security.
- Database monitoring tools.
- Data quality monitoring tools.
- Application mechanisms for helping to ensure accurate data input.

Data - Data Architecture

Principle 4.00.07 Separate the data sources for online transaction processing (OLTP) data and online analytical processing (OLAP) information.

Rationale:

- Separate data sources isolate OLTP systems, which perform mission critical business processing, from large ad hoc queries and online analytical data processing. If the data

sources are not separate, ad hoc queries and direct access of data for OLAP systems can adversely impact online transactional processing.
- Data design is adapted for optimal performance for each type of application, OLTP or OLAP. For optimal performance, OLTP and OLAP may require different database designs. OLAP typically includes complex operations involving time series, dimensional, and trend analysis, which do not perform well with relational database technology alone (e.g., sometimes other methods of data storage are needed to support OLAP, such as multi-dimensional databases or flat files). Note: For more information about OLAP, refer to the Data Warehouse Architecture chapter.

Data - Data Architecture

Principle 4.00.08 Information is one of the most valuable assets for making business decisions.

Rationale:

- The successful delivery of government services depends on conclusions derived from accurate, timely, well maintained, and secure information.
- The value of information is proportional to its availability. If information is not accessible or is too hard to use, it is of questionable value.
- Information used frequently or used by many organizations is extremely valuable.

Data - Data Architecture

Principle 4.00.09 The role of the data warehouse is to accelerate the business decision making process.

Rationale:

- New legislation is constantly redefining the services provided by the state. An accelerated decision making process is required, using timely, easily accessible, understandable, reliable, and high quality information.

Data - Data Architecture

Principle 4.00.10 Data warehouse efforts need support from all levels of the business organization, from the business end user to the management level.

Rationale:

- Initial data warehouse investments can be significant, but the return on investment (ROI) is worth it.
- Business end users need to be involved in the data warehouse effort from the beginning.

Data - Data Architecture

Principle 4.00.11 In general, there is no new data, but there is new information. Existing data from multiple sources is being transformed into intelligent and proactive information.

Rationale:

- How the data is used is far more important than the data itself (i.e., even if data is accurate, it can still be used ineffectively).
- The most effective use of data is to turn it into proactive information that responds to business events (e.g., the "push model" of information).

Data - Data Architecture

Principle 4.00.12 Business units are demanding faster access to more information (not just data).

Rationale:

- Data warehouses are most successful when they are built to support business user demands.
- The growing need for data warehouses is an enabling force for implementing enterprise N-tier client/server systems. N-tier client/server systems segregate data from application processes and make it available for use by other applications.

Data - Data Architecture

Principle 4.00.13 Decision-makers should not be overwhelmed with an excessive volume of unnecessary information.

Rationale:

- Too much information gets in the way of focusing on the most important issues that arise at a specific point in time. Some data warehouse efforts put any data in a data warehouse that may be useful in the future, not just the information to assist in a business need. If there is much more information than is needed, it can overwhelm an end user rather than answering their specific decision support need.
- Information that supports OLAP analysis should be proactively presented to business users. Information in a data warehouse can be presented to business end users so that they know it is available and they can use it.

Data - Data Architecture

Principle 4.00.14 A major source of application developer productivity begins in the data warehouse.

Rationale:

- If a data warehouse is properly implemented, end users can perform their own ad hoc queries and reports against the data warehouse. Application developers do not have to develop programs to anticipate and address each reporting need.

Data - Data Architecture

Principle 4.00.15 A data warehouse will continue to evolve over the life of the business.

Rationale:

- Business and information needs are constantly growing and changing, requiring the addition or modification of data stored in a data warehouse.
- Changes in the use of or in the design of the warehouse should be anticipated and expected.

Data - Data Architecture

Principle 4.00.16 Online transaction processing (OLTP) databases and online analytical processing (OLAP) information databases should have separate data storage areas.

Rationale:

- Separate data storage isolates OLTP systems, which perform mission critical business processing, from large ad hoc queries and online analytical data processing. If data storage is not separate, ad hoc queries and direct access of data for OLAP systems can adversely impact online transactional processing.
- Data design for each type of application, OLTP or OLAP, can be optimized for performance. OLTP and OLAP may require different database designs. OLAP typically includes complex operations involving time series and trend analysis, which do not perform well with relational database technology alone (e.g., sometimes other methods of data storage are needed to support OLAP, such as multi-dimensional databases or flat files).
- For more information about OLTP data, refer to the Data Architecture chapter.

Data - Federated Metadata

Standard 4.01.01 Conform to N.C. GILS standards.

Rationale:

- The U.S. government created a Government Information Locator Service (GILS) to assist the public in locating information from federal agencies. This service is an electronic "card catalog," which describes and indexes information resources of many types.
- Content standards and search and retrieval protocols for US GILS are governed by the "Application Profile for GILS", version 2. The GILS profile establishes a set of attributes for describing and indexing information in a standard way. Further, the profile incorporates portions of the ANSI/NISO Z39.50 Search and Retrieval Protocol (also called ISO23950). Z39.50 is an open communications protocol for searching and retrieving electronic information over a network. Its use enables uniform access to a large number of diverse information sources over the Internet.
- Several states and foreign countries have adopted the GILS standard, making it the foundation of a Global Information Locator Service.
- As a result of Executive Order 100 (September 12, 1996) from Governor Hunt and to meet the needs of agencies to develop diverse types of metadata without duplication of effort, North Carolina has established a government information locator service compliant with the federal GILS. North Carolina's implementation of GILS is called NC GILS.

- State and local agencies in North Carolina are required by the Public Records Law (North Carolina G.S. § 132-6.1 (b)) to index all databases created or significantly modified. NC GILS provides a standard way for agencies to comply with the amended public records law when indexing their databases. Information about databases indexed through NC GILS will be linked to other indexes developed by the federal government, state and local governments, and foreign countries, thus enabling global exchange of public information.
- A central gateway for searching all NC GILS indexes is available on the Internet at http://www.findnc.net or http://www.findnc.org. For more information about NC GILS, refer to the "Guidelines for Agencies Using the North Carolina Government Information Locator Service (NC GILS)." The guidelines are available on the NC GILS Development Headquarters Web site at http://www.ncgils.state.nc.us/.

Data - Federated Metadata

Best Practice 4.01.01 Use and actively maintain the Federated Metadata Repository to store federated metadata definitions.

Rationale:

- Storing data element definitions in a central repository incrementally builds the enterprise data model.
- The repository must be actively maintained (e.g., changes to metadata occur in the repository before the changes occur in operational applications).
- The repository serves as a centralized data administration tool and helps promote data reusability, reliability, and sharing across the enterprise.

Data - Federated Metadata

Best Practice 4.01.02 When designing or modifying a database, review the Federated Metadata Repository for existing standard and proposed data elements before implementing a new database to ensure data elements are defined according to FMR standards.

Rationale:

Design reviews are essential to ensure that shared statewide data is defined consistently across all applications. Design reviews also determine whether data that already exists is consistently defined and not redundantly stored.
Design reviews should document the following:
- Where is this application getting its data?
- What other applications are getting data from this application?
- Is data used by this application defined consistently with statewide definitions? If not, is there a plan to define the data according to enterprise definitions?
- A design review evaluates the data requirements of a project and identifies the following:
- A data requirement that can be solved by using existing federated metadata element.
- Data not already identified as federated metadata must be proposed as an inter-agency or statewide standard to the Metadata Element Review Team to become federated metadata.

- Access is available for application development projects to reference the FMR in order to actively research data requirements. Review the existing standard and proposed data elements in the FMR before implementing a new database to ensure data elements are defined according to standards.
- Key information about data is stored in the systems that are already implemented in the state. If possible, evaluate existing systems to propose statewide and agency data elements.

Data - Federated Metadata

Standard 4.01.02 Conform to the N.C. Public Records Law.

Rationale:

- Any database that falls under the Public Records Law (North Carolina G.S. § 132-6.1 (b)) must be indexed according to the guidelines established by the N.C. Division of Archives and History and published as Public Database Indexing Guidelines and Recommendations. A copy of this publication can be obtained by calling (919) 733-3540 or at http://www.ah.dcr.state.nc.us/e-records/default.htm.
- For more information regarding the retention and destruction of any electronic records, refer to the web site http://www.ah.dcr.state.nc.us/e-records/default.htm or contact the State Records Center staff at (919) 733-3540.

Data - Federated Metadata

Standard 4.01.03 Populate the Federated Metadata Repository with database information for any project requiring IRMC certification (both custom systems and commercial off-the-shelf systems).

Rationale:

- By submitting the appropriate database information to the FMR, an application will automatically comply with both the NC Public Records Law and NC GILS.
- Established federated definitions facilitate collaboration.
- Refer to: http://www.federated.its.state.nc.us/ for more information about what database-related information must be defined in the Federated Metadata Repository.

Data - Federated Metadata

Best Practice 4.01.03 Define existing databases in the Federated Metadata Repository.

Rationale:

- If possible, existing databases should be defined into the database component of the FMR.
- Centralized data management is crucial to the quality and consistency of shared data and requires a quality assurance and quality control process in place for enterprise data.
- As decentralized databases are implemented across the organization, centralized administration will be
crucial to the quality and consistency of the data. Use of inaccurate and inconsistent data is of questionable value.

Data - Federated Metadata

Standard 4.01.04 Custom systems must comply with existing FMR standard data element definitions.

Rationale:

- New databases belonging to custom systems must comply with FMR element definitions. The Metadata Element Review Team will review the data elements to determine if the elements conform to existing standards.
- Any new potential data element standards must be proposed and reviewed for approval as a state standard.
- If the data element definition cannot be customized to conform to existing standards, a waiver must be requested.

Data - Federated Metadata

Best Practice 4.01.04 Use the FMR and the Metadata Element Review Team to create federated definitions of enterprise level data and encourage the sharing of data across agencies.

Rationale:

- Data used by multiple business units must be commonly understood and consistently referenced by all business users. Enterprise sharing of data can only be achieved by creating federated definitions.
- Federated definitions of data emerge through the context of projects. An enterprise model evolves over time through ongoing projects.
- In order to create federated definitions of data, cooperation is needed among the business data owners.
- Centralized management of distributed enterprise-level data is provided through the state's Federated Metadata Repository. This repository is available through the Internet.
- The state's Technical Architecture and Project Certification Committee of the IRMC currently approves any statewide and inter-agency standards.

Data - Federated Metadata

Standard 4.01.05 Commericial off-the-shelf (COTS) systems that support client-controlled data element definitions must comply with FMR standard for data element definitions. Or, vendor must provide conversion routine that meets metadata exchange standards for data sharing

Rationale:

- If an off-the-shelf system has data formats that can be modified, the data elements should be adapted to conform to the standard data requirements.
- If the data element definition cannot be customized to conform to existing standards, the vendor must provide conversion routines to conform to the FMR metadata exchange standards.

Data - Federated Metadata

Best Practice 4.01.05 Identify authoritative sources for federated metadata.

Rationale:

- Authoritative business sources for federated metadata must be identified, documented, and actively maintained in the repository. Authoritative business sources are the business units responsible for the accuracy of the data stored.
- A source of record is an authoritative source for data. Data in a source of record is trusted to be accurate and up-to-date. All other data stores should synchronize to the source of record. The data in record sources must be actively managed and the data model should be verified by data administrators. Tools and quality control techniques must be applied to the contents of the data stores themselves, in order to ensure the -quality of the data.
- Each application must identify data sources for all data that it does not originally capture. The application capturing the original data is the authoritative source, and is responsible for the quality of the data. All application data models for ongoing projects should be reviewed to ensure that data existing in authoritative systems is reused and not redundantly stored.

Data - Federated Metadata

Standard 4.01.06 Use Federated Metadata Exchange Standards when exchanging data across agencies.

Rationale:

- If data needs to be exchanged across agency boundaries and the data is physically stored differently, then the data must be exchanged through the exchange standard as specified in the FMR.
- For more information about sharing data, refer to the Data Access Implementation topic in this chapter.

Data - Data Modeling

Best Practice 4.02.01 Determine the initial business requirements using terms and tools that are familiar to business users.

Rationale:

- For example, use a simple tool like a spreadsheet containing familiar business terms for the data.
- Once a simple high-level view is designed, data modeling tools can be used internally to develop a more detailed data model.

Data - Data Modeling

Standard 4.02.01 Comply with Federated Metadata Repository standard data element definitions.

Rationale:

For more information about the Federated Metadata Repository, refer to the Federated Metadata topic in this chapter.

Data - Data Modeling

Best Practice 4.02.02 Take the Entity-Relation (ER) model to the third normal form, then denormalize where necessary for performance.

Rationale:

- The third normal form is the most commonly recommended form for the ER model.
- In some cases, a denormalized database can perform faster as there can be fewer joins, or reduced access to multiple tables. This process saves both physical and logical input and output requirements.

Data - Data Modeling

Best Practice 4.02.03 In a dimensional model, use a star schema whenever possible.

Rationale:

Use a snowflake schema only if it increases user understandability or improves performance. A snowflake schema may add unnecessary complexity to the data model.

Data - Data Modeling

Best Practice 4.02.04 Restrict free form data entry where possible.

Rationale:

- In the design phase, consider the values that may be input into a field. These values or domains should be normalized so that data is consistent across records or instances. For example, using consistent values for gender or address information.
- Use look-up tables and automate data entry for column or attribute domain values to restrict what is entered in a column.

Data - Data Modeling

Best Practice 4.02.05 Setup indexes and form relationships carefully.

Rationale:

- Limit the number of indexes on databases that will be experiencing significant insert and update activity. When an insert is performed, not only is the record updated, but all the indexes are updated as well.
- Increase the number of indexes on databases where importance lies in retrieval time. Indexes can increase performance on retrieval time.
- Before creating a database, indexes, or data access programs, verify that all relationships have been documented.

Data - Data Modeling

Best Practice 4.02.06 Design the data model to allow for growth and/or change.

Rationale:

Design data models to accommodate any future changes, including growth and changes in business requirements or database technologies.

Data - Data Modeling

Best Practice 4.02.07 Archive and protect the data model.

Rationale:

- Data models store a wealth of agency and statewide information and must be archived and protected.
- Data models should be catalogued with the agency's project documentation and used to facilitate future revisions.

Data - Data Modeling

Best Practice 4.02.08 Each agency should standardize on a common data modeling tool for designing and maintaining all new database instances.

Rationale:

- Agency Technical Architectures specify each agency's common data modeling tool.
- A data model ensures that data is defined accurately so it is used in the manner intended by both end users and remote applications. For more information about data modeling, refer to the Data Modeling topic in this chapter.

Data - Data Modeling

Best Practice 4.02.09 Use a data modeling tool to reverse engineer existing databases.

Rationale:

Data modeling tools can evaluate an existing database structure and reverse engineer a data model. The reverse engineered data model can be used to capture valuable information about the existing database.

Data - Data Base Management System

Standard 4.03.01 New databases must use a relational DBMS supporting ANSI-standard SQL (currently SQL92).

Rationale:

- Relational databases offer dependability, flexibility, and compatibility for future data needs.
- Data can be maintained and readily accessed through ANSI-standard SQL calls. SQL is an industry standard for the data access tier of an application and for data access tools.
- Use of proprietary extensions creates vendor lock-in.
- Desktop database products, such as Microsoft Access and FoxPro, are considered end user database access tools and must not be used for agency or statewide implementation.

- Non-relational technology such as flat files can be used for temporary work storage and unstructured data such as textual data. Large-scale use of non-relational technologies must obtain a waiver.
- Use of ODBMS should not be used since the object defines each data type, data is stored in a proprietary format and the data cannot be understood without the object code that describes it. If ODBMS is to be used, a waiver must be requested.

Data - Data Base Management System

Best Practice 4.03.01 When using object-oriented programming languages, use a relational database management system (RDBMS).

Rationale:

- Although there is cost and effort associated with an object-relational model, the relational data model is much more adaptive and understandable. ODBMS only allows access to the encapsulated data defined in the source code. RDBMS allows new relationships and data mappings to be easily defined.
- As applications and associated databases age, applications tend to depreciate because business needs change over time, while data and databases appreciate because of the valuable information contained within.

Data - Data Base Management System

Best Practice 4.03.02 When using a relational database with object-oriented programming, design the relational data model first.

Rationale:

- Object models are typically more complex than the relational model.
- If the object model is built first, building the relational model that matches it may not be possible.
- For more information about relational data modeling, refer to the Data Modeling topic in this chapter.

Data - Data Base Management System

Best Practice 4.03.03 When creating an object-relational mapping between the object model and the RDBMS, keep it simple.

Rationale:

- Simple relationship mappings between objects and relational databases provide ease of use and better performance.
- Use the primary and foreign key relationships in the RDBMS to assist in mapping relationships between objects.

Data - Data Base Management System

Best Practice 4.03.04 Replicate stable data, based on business and performance requirements.

Rationale:

- Replication should not be used unless it is required for performance or decision support.
- A replication infrastructure is simpler to design for stable data. If replicated data is updated frequently (i.e., not stable), it is much more difficult to design and maintain a replication infrastructure.
- For more information about replication, refer to the Data Replication Tools topic in the Data Warehouse Architecture chapter.

Data - Data Access Middleware

Best Practice 4.04.01 Provide centralized administration for data access middleware through central IT staff.

Rationale:

- Typically, workstations and servers are used for multiple applications and require connectivity to multiple databases. If administration for data access middleware is provided by central IT staff, any changes that are required are more easily managed and executed.
- Support costs and efforts to support data access middleware are reduced.

Data - Data Access Middleware

Best Practice 4.04.02 Avoid use of extensions that create vendor lock-in.

Rationale:

- To differentiate their database from other vendors, many database vendors have implemented special extensions beyond the SQL-compliant commands. Although sometimes these extensions may be useful for a particular function, they are not recommended.

Data - Data Access Implementation

Best Practice 4.05.01 Establish a data infrastructure that can accommodate rapid changes in data models based on changes in business requirements or changes in database technologies.

Rationale:

- Business requirements change frequently. The data infrastructure and design must be adaptive and allow for changes to be easily implemented.
- Technology changes are fast emerging. The infrastructure must allow for replacement of the database technology if necessary.

Data - Data Access Implementation

Standard 4.05.01 Use the North Carolina Service Broker (NCSB) for inter-agency data sharing.

Rationale:

- NCSB is the standard for inter-agency data sharing. Inter-agency services deployed using NCSB can be easily leveraged by other authorized applications.

- The agency owning the data is responsible for writing the shared service to access the data. This ensures data integrity and proper data interpretation.
- The agency requesting the data is responsible for writing the request to retrieve shared data according to the shared service specifications.

Data - Data Access Implementation

Best Practice 4.05.02 Centralize data that needs to be shared and current.

Rationale:

- High-volume transaction data that is shared across locations and that needs to be current for all locations must be centralized so all locations have access to the same data source.
- Replicating frequent updates to distributed databases increases systems complexity and network traffic.
- Data must be centralized when one or more of the following criteria occur:
- Many users need access to latest data (i.e., OLTP systems).
- The number of users is small and there are no distributed sites.
- There is a lack of skills and tools at multiple sites to manage distributed data.
- There is a need to provide a consolidated and integrated database for federated metadata on an open platform.

Data - Data Access Implementation

Standard 4.05.02 Use the industry standard of ANSI Standard SQL (currently SQL92) when accessing relational databases.

Rationale:

When using a database access tool that uses SQL calls, do not use any vendor specific extensions.

Data - Data Access Implementation

Best Practice 4.05.03 Design databases to be modular, business driven and aligned with application services, not monolithic.

Rationale:

- Aligning data with the application service facilitates changes in business processes. Only the data associated with a particular business process is potentially affected when a change is needed, not all the data associated with an entire application. It also increases performance for backup and recovery and provides higher reliability, availability, and scalability.
- By modularizing data, this practice provides better performance for backup and recovery, higher reliability, availability, and scalability, and better transaction performance due to parallelism (e.g., a complex request can be broken down and be processed by multiple databases at the same time).
- Very large databases (VLDBs) typically use a terabyte or more of storage. It is recommended that VLDBs be partitioned based on the appropriate business elements to improve application and database performance. VLDB partitioning can enable more

efficient backup and recovery capabilities (e.g., it is more efficient to backup five 10 million row tables simultaneously than it is to backup one 500 million row table).

In order to align data with application services, the following items need to be defined:
- Business processes.
- Data required to service the business processes.
- Business units responsible for providing the business process.
- Access to the data, and the know-how to use the data, by other business units or applications.

Data - Data Access Implementation

Best Practice 4.05.04 Protect data through data access rules.

Rationale:

- There should be no direct access to data. The only access should be through data access rules that own the data. This practice helps to protect data from unauthorized or accidental access.
- For more information on applications, refer to the Application Architecture chapter.
- For more information about data modeling, refer to the Data Modeling topic in this chapter.

Data - Data Access Implementation

Best Practice 4.05.05 Validate data at every practical level to ensure data quality and avoid unnecessary network traffic.

Rationale:

- Validation can be coded into multiple tiers of the n-tier architecture to ensure that only valid data is processed and sent across the network. For example, an invalid field entered in a data entry form can be corrected before data is written to the database.
- Data integrity verification rules should be used when possible.
- For more information about application development, refer to the Application Architecture chapter.

Data - Data Access Implementation

Best Practice 4.05.06 Minimize the replication of data within operational applications by replicating only stable data when necessary and based on business requirements.

Rationale:

- It is better to maintain only one version of data whenever possible, particularly for mission critical OLTP systems.
- Replication must not be used unless it is required for performance or decision support.
- A replication infrastructure is simpler to design for stable data. If replicated data is updated frequently, it is much more difficult to design and maintain a replication infrastructure.

- Replication may be appropriate when there are users in different locations needing similar data that does not need to be current 24 hours a day and a central source database is not a possible solution. (See Figure 4-22.)
- Specific application requirements for data availability, recoverability, and freshness (near real time, 24 hours old, etc.) must be identified.
Long binary or text value

Data - Data Access Implementation

Best Practice 4.05.07 Design for all replicated data to be read only.

Rationale:

Updates must be directed to the authoritative source, not the replicated data.

Data - Data Access Implementation

Best Practice 4.05.08 Perform all data updates to the authoritative sources, then replicate changes to remote databases.

Rationale:

- An authoritative source for data is the source of record where data is collected and maintained by the application that owns the data. All other data stores must synchronize to the source of record. All data updates must occur against the source of record through the data access rules that own that data. (See Figure 4-23.)
- The N-tier Application Architecture facilitates the implementation of reusable data access rules.
For more information on accessing data stored in legacy systems, refer to the Integration Architecture chapter.
Long binary or text value

Data - Data Access Implementation

Best Practice 4.05.09 When replication is needed, evaluate the replication options and select the implementation that meets the existing business needs.

Rationale:

The following considerations must be made:
- Acceptable lag times must be defined to determine replication schemes, schedules, and the product features needed.
- Replication requirements must be defined to determine if the replication tools are sufficient.
- Business requirements must be documented for any data transformation requirements and for any differences in replication requirements.
- How the impact of processing overhead on the source and target databases will affect system performance.
- How network traffic may impact communication costs and performance.
- Tools providing capability for configuration, monitoring, tuning, and administration should be analyzed.

Data - Data Access Implementation

Best Practice 4.05.10 Design the data access infrastructure to support the transparency of the location and access of data by each application.

Rationale:

- This means designing an N-tier architecture where all data access is managed through a middle tier. This design makes databases easy to relocate, restructure, or re-platform the back end services with minimal disruption to the applications that use them. It is essential for adaptive systems.
- For more information on N-tier architecture partitioning, refer to the Conceptual and Application Architecture chapters. For more information on communicating between application tiers, refer to the Application Communication Middleware Architecture chapter.
- A client should not send SQL requests directly to a server. Instead of using SQL code, the client should communicate with the database through data access rules. The application receives a request from a client and sends a message to the data access rule. The data access rule sends an SQL call to the database. With this method, the client does not send SQL to the server, it sends a request for work. (See Figure 4-24.)
Long binary or text value

Data - Data Access Implementation

Best Practice 4.05.11 Design for data to be accessed only by the programs and business rules owning the data, never by direct access to the database.

Rationale:

- This practice ensures security, data integrity and accurate interpretation of the data and allows for adaptability to changes in business needs.
- For more information on accessing the programs that own legacy data, refer to the Application Integration topic of the Integration Architecture chapter.

Data - Data Access Implementation

Best Practice 4.05.12 For data quality management, implement tools, methods, processes and policies to provide high-level data accuracy and consistency across distributed platforms.

Rationale:

- Both business users and Information Technology (IT) staff are responsible for data accuracy and consistency. Policies and procedures must be established to ensure the accuracy of data.
- IT staff is responsible for and must provide security mechanisms to safeguard all data under IT control. The business users must determine functional security requirements, while the physical security must be provided by IT.
- Applied systems management provides safeguards against data loss and corruption and provides the means of recovering data after system failures. This implies that effective backup and recovery systems are imperative and that data can be recovered in a timely basis regardless of the cause of loss.

- To satisfy service-level requirements, if the data is not too volatile, data replication can be used. If the data is extremely volatile, using replicas must be avoided. Additional requirements may include providing redundancy for extra bandwidth for communications volume and for availability in the event of a disaster.
For critical functions, plan for survivability under both normal operations and degraded operations. (For more information on disaster recovery, refer to the Enterprise Systems Management of IT Assets chapter.)
- It would be ideal to record the flow of data across systems, even if it is built incrementally starting with existing application development projects.

Data - Data Access Implementation

Best Practice 4.05.13 Optimize the physical database design to support an optimal total performance solution.

Rationale:

As with all application and data access design, performance is always a factor to consider. However, database performance is only part of the total solution and must be evaluated in conjunction with other components that impact performance, such as network and application. When implementing data access, several practices can help performance, including:
- Limit the number of indexes in a database. When a record update occurs, not only the record is updated, but all the indexes are updated as well.
- Limit ad hoc data access. End user ad hoc access can impact the performance of the database.
- Limit the number of rows returned in a query. In OLTP, most users normally work with only a single row at a time or a few rows displayed in a grid, list or combo box. If a user will only be working with a handful, there is no reason to return all the rows in a table.
- Return only the columns needed. Provide an explicit column list instead of a 'SELECT *" query.
- Limit the number of joins. Complex multi-table joins have negative performance ramifications.
- Avoid sorts. Sorts can be slow, especially if sorting large amounts of data. If sorting is required, sort on indexed fields.
- Limit the rows used for pick lists, combo boxes, or lookup tables. If a large list is necessary, find an alternate method to provide the list.

Data - Data Access Implementation

Best Practice 4.05.14 Implement a minimal number of data access rules.

Rationale:

- A typical n-tier application has numerous business rules. The data access logic for these business rules should be shared through a minimal number of reusable data access rules. Due to the commonality of database queries, many similar queries can execute using a single, properly planned data access routine. (See Figure 4-25.)
- Portability to another database platform or vendor is simplified by having a fewer data access rules.

Long binary or text value

Data - Data Access Implementation

Best Practice 4.05.15 Use ANSI-Standard SQL programming language to access a database.

Rationale:

Data access to relational data stores must be through ANSI-standard SQL programming language access, not proprietary SQL extensions.

Data - Data Access Implementation

Best Practice 4.05.16 Implement a minimal amount of data access rules stored in the database as stored procedures and triggers to avoid vendor lock-in.

Rationale:

- Code data access rules into a data access service. When implemented through the North Carolina Service Broker (NCSB), these services are callable by multiple applications. If a database changes, there is minimal impact to the calling applications since the API should not change.
- Stored procedures and triggers are specific to the database vendor and are more difficult to migrate to a view database if required.
- Database triggers must only be used to support referential integrity.
- Other technologies such as object transaction monitor (OTM) can be used to negate the impact of executing dynamic SQL.

Data - Data Access Implementation

Best Practice 4.05.17 Use the North Carolina Service Broker (NCSB) for intra-agency and intra-application data sharing.

Rationale:

NCSB is already the standard for inter-agency data sharing. By using the NCSB for intra-agency and intra-application data access, the service can easily be adapted if other authorized applications require data sharing.
The agency owning the data is responsible for writing the shared service to access the data. This ensures data integrity and proper data interpretation.
The agency requesting the data is responsible for writing the request to retrieve shared data according to the shared service specifications.

Data - Data Access Implementation

Best Practice 4.05.18 Use the state's interface engine for data sharing of legacy platform data or other data where the application source code cannot be modified or interfaced.

Rationale:

- Some legacy systems do not have an application program interface (API) and there is no access to the source code. When requiring data access to one of these systems, use the state's interface engine.
- The state's interface engine can be used in instances where the source code is unable to be modified or the data layouts are unavailable. It is an unobtrusive interface to accomplish data sharing.
- Use of database gateway or database-specific middleware must be avoided.
- For more information about the state's interface engine, refer to the Integration Architecture.

Data - Data Security

Best Practice 4.06.01 Perform a risk assessment for the application database and data elements to determine level of security required.

Rationale:

- To assure adequate protection of data assets, perform a risk assessment to identify specific security concerns that must be addressed before development and deployment of an application.
- The security analysis will determine what measures must be put in place to restrict end users and applications from viewing, modifying, or deleting confidential or private data. The security analysis may reveal that adequate measures are in place to restrict end users and applications from viewing, modifying,
and deleting low impact and public data.
- Classify users according to their functional data needs (e.g., outside access from business partners, citizens, suppliers, etc.)

Data - Data Security

Standard 4.06.01 Change all default database passwords (i.e., sa accounts, etc.).

Rationale:

- System administrator accounts have full access to all databases in a database server. Hackers often attempt a login to a system administrator account using a default password. As soon as a database is set up, change all default passwords.

Data - Data Security

Best Practice 4.06.02 Use generic, protected user accounts for direct database access to streamline administration, ensure scalability, and protect against non-application data access.

Rationale:

- When a generic, protected user account is used, each individual user account is not defined to the database, so end users are unable to gain ad hoc access to the data. Their only access should be through the application.
- The individual user account is only defined at the application level, and does not have to be maintained in more than one place.

- Implementing generic users makes applications scaleable since each process is not tied to a specific user.
-The generic user account and password used to access data in the back end must not be protected and not accessible to end users.

Data - Data Security

Best Practice 4.06.03 Implement data security to allow for changes in technology and business needs.

Rationale:

- Implement security to be a roadblock to unauthorized access, but not a hindrance to access by authorized users. Implement the minimal number of sign-on or authentication processes if possible.
- An adaptable security infrastructure must be implemented to allow for changes in technology, business needs, and reactions to intrusions.
- Monitor ITS and industry security alerts and recommendations. Security tools and techniques are rapidly changing and enhancements are being made. Monitor the industry and ITS recommendations and implement changes to security configurations as needed.

Data - Data Security

Best Practice 4.06.04 Handle sensitive data carefully.

Rationale:

- Confidential or private data must not be stored on a laptop without password protection or encryption. Laptops are vulnerable to loss of data through hackers, thieves, and accidents. Sensitive data must be secured on a database server with proper policies and procedures in place to protect the data.
- Ensure that passwords are encrypted both inside application executables and across the transport layer.
- Password and data encryption in databases and laptops can be provided by third party products.
- A backup and recovery plan for databases and laptops must be in place

Data - Data Security

Best Practice 4.06.05 Provide measures for laptops to backup their data, like zip drives, etc.

Rationale:

- Only non-sensitive data should be stored on a laptop. If possible, the authoritative source must be on a server, and data should be replicated to the laptop.
- When data is stored on a laptop, provide easy-to-use backup facilities. Implement policies to ensure and automate backup

Data - Data Security

Best Practice 4.06.06 Record information about users and their connections as they update and delete data. Auditing can determine who updated a record and their connection data.

Rationale:

The information that can be captured by the application includes:
- The user account the user logged in with.
- The TCP/IP address the connected user's workstation.
- The certificate information (if using certificates) about that user.
- The old values that were stored in the record(s) before the modification.
- The new values that were input to the record(s).

Data - Data Security

Best Practice 4.06.07 Implement transaction logging so recovery of original data is possible and protect the transaction log.

Rationale:

- Transaction logging records activity on the database and can be used to roll back a transaction.
- Protect the transaction log through access control and backup. Only the database should be writing to the transaction log. All other access should be read only.
- The transaction log should be located on a separate physical disk if possible. If not possible, use RAID to protect the integrity of the log file.

Data - Data Security

Best Practice 4.06.08 Implement security scanning and intrusion detection at the database level if possible.

Rationale:

- Scan the database and database server for potential weaknesses before they become a problem. Implement any recommendations of the security management tool. For example, a tool may advise to disable FTP services on a database server.
- Monitor the database for possible intrusions. For example, monitor and alert when multiple invalid login attempts occur. Intrusion detection protects the database server from attacks from both sides of the firewall e.g., internal network, WAN, or Internet).
- Audit logins, user account creation, and failed login attempts.

Data - Data Security

Best Practice 4.06.09 Ensure data integrity by securing data movement or data transport.

Rationale:

- When high impact, sensitive data is transported through the LAN, WAN, or Internet, ensure that the data is encrypted and protected from alterations. This can be accomplished through Secured Socket Layers (SSL) or Virtual Private Network (VPN).

- Other types of data must be encrypted and protected if there is a risk of the data being altered.

Data - Data Security

Best Practice 4.06.10 Protect database servers from hardware failures and physical OS attacks.

Rationale:

- Database servers must be located in a climate-controlled, restricted-access facility, and preferably a fully staffed data center. Uninterruptible power supplies (UPSs), redundant disks, fans, and power supplies must be used.
- For more information about hardware, refer to the Platform Architecture chapter.

Data - Data Security

Best Practice 4.06.11 Protect source code in data access rules, particularly if it contains password information.

Rationale:

- On the back end, an application needs to store account and password information in order to authenticate to a database or other application service. Protect the source code from unauthorized viewing.
- Store passwords in an encrypted format when possible.

Data - Data Security

Best Practice 4.06.12 Do not store credit card numbers in the database for non-recurring charges or infrequent recurring charges. Store authorization numbers and discard credit card numbers after use.

Rationale:

- For infrequent recurring charges (for example an annual fee), or non-recurring charges (for example a one- time fee), storing credit card numbers and expiration dates in a database, even encrypted, can present an unjustifiable risk for the state.
- A credit card number is only necessary to request authorization. Keep the credit card number only until
authorization is complete, then discard. The authorization number can be used to track activity and verify authorization.

Data - Data Security

Best Practice 4.06.13 Protect and encrypt credit card numbers when storing for recurring charges. Store personal verification information independently.

Rationale:

- Certain business requirements, such as frequent recurring charges, may require credit card numbers to be stored in a database.

- When it is absolutely necessary to store credit card numbers, encrypt the credit card number in the
database. To further protect the credit card, store personal verification information, such as name and address, in a separate database from credit card information. Use different user accounts for each database connection.

Data - Data Warehouse

Best Practice 4.07.01 Do not wait for an enterprise information model to start data warehouse efforts.

Rationale:

Enterprise models often take years to complete. Meanwhile, business requirements are changing rapidly. Data warehouse efforts begin with a strategic set of information, preferably targeting a cross section of users.
Develop the data warehouse incrementally:
- Start small. Start by putting a relatively small amount of strategic data on a separate server, solving the problems of a specific set of users. The first step should be measured in time rather than gigabytes. The first stage should be no more than 90 days, from initiation to implementation.
- Once the data is ready, it is time to add on-line analytical processing (OLAP) capability. For complex decision support, add on-line analytical processing, including trend analysis, indexing technology, and multi-dimensional database technology.
- Implement near-real time production data feeds to keep the database populated.
- Apply data extraction technology to accomplish real time feeds to maintain the data. (Note: For real time data feeds, the multidimensional indexing engine must be a separate database because it cannot accommodate real-time feeds).

Data - Data Warehouse

Best Practice 4.07.02 Begin data warehouse efforts by addressing a specific requirement for a specific decision support application, keeping growth and scalability in mind.

Rationale:

- This practice is similar to data mart design, but the tools and databases used should be designed to support a large data warehouse.
- Use vendor supplied products designed to support a large data warehouse. Vendor-supplied data mart tools are not typically scaleable to support the migration from a data mart to a data warehouse solution. These tools are designed to quickly implement a specific solution.

Data - Data Warehouse

Best Practice 4.07.03 Identify specific requirements for data availability, freshness (i.e., live, 24 hours old, etc.), and recoverability.

Rationale:

- Some data warehouses need to be updated more frequently than others. When the original data is frequently changing or is more volatile, it may be necessary to update the data warehouse on a near real time basis.
- On the other hand, if the original data is fairly stable and not as volatile, the data warehouse may only need daily, weekly, or even monthly updates. For example, a data warehouse that stores criminal data contains more volatile information and needs to be updated more frequently than a data warehouse that stores state registered corporation name and address information for public access.

Data - Data Warehouse

Best Practice 4.07.04 Perform benchmarks on the database design before constructing the database.

Rationale:

- Expect to make changes and adjustments throughout development.

- Changes during the early cycles up to, and including implementation, are a primary mechanism of performance tuning.

Data - Data Warehouse

Best Practice 4.07.05 Normalize application access to all decision sources.

Rationale:

- Common functions for data access, such as a "Compute Age" function, should be reusable and shareable by multiple application systems.
- For more information about shared business rules and components, refer to the Application and Componentware Architecture chapters.
Long binary or text value

Data - Data Warehouse

Best Practice 4.07.06 Ensure that the appropriate security is applied to the data warehouse.

Rationale:

- Prevent unauthorized users from tampering with the data.
- Control access to portions of the database on a user-by-user basis. Certain types of data that is stored in a data warehouse may be sensitive. Providing protection for privacy of individual and confidentiality of data must be considered.

Data - Data Warehouse

Best Practice 4.07.07 Choose a data warehouse project manager who is user-oriented rather than technology oriented.

Rationale:

- A user-oriented manager ensures that the data warehouse will meet the business needs of the end users.
- The data warehouse project manager must manage the expectations and sponsorship of the data warehouse.
- The data warehouse manager can make sure the data is easy to use and understand.

Data - Data Warehouse

Best Practice 4.07.08 Allow only read only access to end users of data warehouses.

Rationale:

- Updates should only occur to the operational (OLTP) source where the data originates.

Data - Data Warehouse

Best Practice 4.07.09 Direct all information queries against decision support databases, not OLTP databases. Conversely, operational transactions should be directed to operational databases only, not OLAP databases.

Rationale:

- Data warehouses, and data marts contain data that has been checked for consistency and integrity, and represents a cross-functional view of data.
- Data in transaction (OLTP) systems typically support a specific business group or function.
- OLTP transactions should not depend on a data warehouse database. They require a stable operational environment that is not affected by ad hoc usage or external data.

Data - Data Warehouse

Best Practice 4.07.10 Establish the data warehouse as the authoritative source for all other decision support databases.

Rationale:

- Data administration policies, procedures and tools are required.
- Project design reviews are required. A mandatory deliverable must be identification of all data sources for the project.
- Distributed data warehouse servers and data marts should use an authoritative source for data feeds.

Data - Data Warehouse

Best Practice 4.07.11 Store atomic-level data in the data warehouse in addition to summary data.

Rationale:

- Atomic data is transaction-level data. It contains much more detail than summary data.
- Atomic-level data addresses the business need to recast history. Due to the fast pace of business change, many organizations are going through multiple reorganizations. After a reorganization, many decision makers want to recast history (e.g., to get a feel for what

test scores would have been like if the number of school districts was already reduced to respond to legislation or funding).
- If only summary level historical data is kept in the data warehouse, it is not possible to recast history.

Data - Data Warehouse

Best Practice 4.07.12 Perform periodic validity audits against the data warehouse information model to ensure a high level of confidence in the quality and integrity of the data.

Rationale:

- Accelerated decision making requires high quality data. If operational data has changed or additional data is needed, changes must be made in the information model and in the data warehouse itself.
- The data stored in a data warehouse should conform to the information model.
- The source data populating a data warehouse should be verified for consistency and accuracy.
- The data warehouse should still correspond to business needs.
- Ensuring the integrity and quality of data is the responsibility of both the business users and IS.

Data - Data Warehouse

Best Practice 4.07.13 The implementation plan should match critical business needs.

Rationale:

- Start with strategic business needs.
  Optimize critical data access before less-critical data access.
  Optimize high-volume data access before low-volume data access.
  Optimize applications and data with high-service-level requirements before those with lower requirements.

Data - Data Warehouse

Best Practice 4.07.14 Consider the network when designing OLAP systems.

Rationale:

- The network is a partner to an application and can impact performance and design.
  Estimate the impact on the network when partitioning data.
  For more information about networks, refer to the Network Architecture chapter.

Data - Data Warehouse

Best Practice 4.07.15 Plan and budget for the ongoing operations, administration, and maintenance of a data warehouse.

Rationale:

- A support plan for the data warehouse should be documented and implemented.

- The data warehouse should be scaleable to meet future demands.

Data - Repository

Best Practice 4.08.01 Maintain a repository for every data warehouse.

Rationale:

- The repository contains metadata, or information about the data, in the data warehouse.
  The repository represents the shared understanding of the organization's data.
  The repository can be built incrementally, in stages, based on data warehouse design and implementation.
  The repository should support multiple types of data elements, such as graphics.

Data - Repository

Best Practice 4.08.02 Actively maintain the repository.

Rationale:

- Changes in the repository must occur before the changes to the data warehouse environment.

Data - Repository

Best Practice 4.08.03 The repository management system used to store metadata for a data warehouse should be built with the same repository tools that are used to manage federated data metadata.

Rationale:

- Federated data metadata reference many common data elements used by multiple application systems. Due to the commonality of federated data, it will frequently be used in a data warehouse environment.
- If the repository databases use the same management systems, metadata will easily transfer between repositories to build the information model.
- For more information about the federated data information repository, refer to the Data Architecture chapter.

Data - Data Hygiene Tools

Best Practice 4.09.01 Use the data warehouse metadata repository to document the rules applying to data scrubbing.

Rationale:

- The information about how the data is to be scrubbed should be saved for historical purposes.

Data - Data Hygiene Tools

Best Practice 4.09.02 Determine the schedule for data cleansing during the design of the data warehouse.

Rationale:

- Depending on the sources of the original data, some data warehouses may need to be cleansed more frequently than others are.

Data - Data Hygiene Tools

Best Practice 4.09.03 Ensure data entry quality is built into new and existing application systems to reduce the risk of inaccurate or misleading data in OLTP systems and to reduce the need for data hygiene.

Rationale:

- Provide well-designed data-entry services that are easy to use (e.g., a GUI front end with selection lists for standard data elements like text descriptions, product numbers, etc.).
- The services should also restrict the values of common elements to conform to data hygiene rules.
- The system should be designed to reject invalid data elements and to assist the end user in correcting the entry.
- All updates to an authoritative source OLTP database should occur using the business rules that own the data, not by direct access to the database.
- Attention to detail should be recognized and rewarded.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.01 During data warehouse design, determine the logic needed to convert the data, plan and generate the extraction and transformation routines, and quality assure the data populating the data warehouse.

Rationale:

- Planning for data extraction and transformation should start at the same time the data warehouse design starts.
- Data extraction and transformation is an important process for populating the data in a data warehouse and for ensuring that the data in a data warehouse is accurate.
- Data extraction and transformation logic includes data conversion requirements and the flow of data from the source operational database to the data warehouse.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.02 Assess the source data that will populate a data warehouse for accuracy and quality.

Rationale:

- Data needs to be accurate to ensure good business decisions.
- Data needs to be relevant to the business need and consistent across multiple sources.
- Data must be complete. It must contain the information necessary to answer the data warehouse business need.
- The data assessment also involves evaluating the business rules associated with that data. The appropriate business rules must be applied to the data to maintain accuracy.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.03 Apply data hygiene techniques to the warehouse data after it has been extracted.

Rationale:

- The data warehouse will contain data from disparate operational data sources. This data will need to be cleaned to resolve any differences before it can be stored for analysis in the data warehouse.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.04 Develop a schedule for data extraction that both meets the needs of the data warehouse users and does not impact an OLTP system.

Rationale:

- Evaluate the impact of data extraction to any OLTP systems accessed.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.05 Document data extraction and transformation information in the data warehouse repository.

Rationale:

- Data extraction and transformation metadata are important aspects of a data warehouse. This metadata provides the information map connecting the data populating a data warehouse with its source operational databases.

Data - Data Extraction and Transformation Tools

Best Practice 4.10.06 If a vendor-supplied extraction and transformation product is selected, it should support the same metadata repository that supports the data warehouse. It should also support the physical data warehouse.

Rationale:

- Select products that are capable of interacting with the metadata repository and the data warehouse.
- Metadata drives the operations of the extraction and transformation tools. If the data warehouse repository is not supported by a vendor-supplied extraction and transformation product, a separate metadata repository must be developed and maintained.

Data - Data Replication Tools

Best Practice 4.11.01 Replicated data should be read-only, except where business practices clearly allow inconsistencies.

Rationale:

- It is easiest to manage data quality and integrity when replicated and distributed data is read only.

- Some business applications require updates to occur against the local database.
- Distributed independent updates require a reconciliation process that may be quite complex.

Data - Data Replication Tools

Best Practice 4.11.02 Replicate stable data, based on business and performance requirements.

Rationale:

- Replication should not be used unless it is required for performance or decision support.
- A replication infrastructure is simpler to design for stable data. If replicated data is updated frequently (i.e., not stable), it is much more difficult to design and maintain a replication infrastructure.

Data - Data Replication Tools

Best Practice 4.11.03 When replication is needed, evaluate the replication options and select the implementation that meets the existing business needs.

Rationale:

- Define the acceptable lag times to determine replication schemes, schedules, and the product features needed.
- Document the business needs for any non-identical replicated copies of data and any data transformation requirements.
- Decide if the replication required is within the capabilities of an existing replication product or determine if external processing is required.

Determine if replication processing overhead on the source and target databases will affect the performance of the applications using either database.

Determine if network traffic required for replication can impact communication costs and performance.

Determine the capability of the tools selected for configuration, monitoring, tuning, and administration.

Data - Business Intelligence Tools

Standard 4.12.01 When accessing relational databases, use the industry standard of ANSI Standard SQL (currently SQL92).

Rationale:

- When using a database access tool that uses SQL calls, do not use any vendor specific extensions.

Data - Business Intelligence Tools

Best Practice 4.12.01 Implement as few database access tools as possible.

Rationale:

- If vendor or business specific database access tools are selected that cannot be used to access databases distributed statewide, then multiple tools must be implemented, supported, and maintained.
- Select a tool that has a high degree of functionality, but is easy to learn and use. The learning curve for end users is reduced if they do not have to learn multiple tools.
- Avoid developing custom applications to perform routine data access functions.
- "Common data services" and "shared data" will increase in coming years.

Support costs and efforts to support database access tools are reduced.

Data - Business Intelligence Tools

Standard 4.12.02 Use ODBC from any data access programs rather than vendor-specific database access tools.

Rationale:

- ODBC allows flexibility in programming. A database can be easily modified or relocated. If a change is needed, the change is made to the ODBC configuration file, not to each business intelligence program or tool.

Data - Business Intelligence Tools

Best Practice 4.12.02 Implement decision support and executive information applications using an N-tier application architecture.

Rationale:

- By developing and application system in N tiers, systems can respond quickly to changes in business needs.
- Decision support and executive information systems application programs benefit from the use of N-tier and reusable and shared components.
- For more information about N-tier application programs and reusable components, refer to the Application Architecture and Componentware chapters.

Data - Business Intelligence Tools

Best Practice 4.12.03 There should be no ad hoc query access to OLTP databases.

Rationale:

- Ad hoc query access to online operational databases can severely impact the performance of mission critical operations. A data warehouse should be implemented for users with ad hoc query needs.

Data - Business Intelligence Tools

Standard 4.12.03 Implement a server-based ODBC solution rather than a workstation-based ODBC implementation.

Rationale:

- A server-based ODBC solution is easier to administer. ODBC database changes and additions are easier to manage, since updates are made to ODBC servers, not every workstation that uses ODBC.

Data - Business Intelligence Tools

Standard 4.12.04 Use domain name system (DNS) names for databases that are accessible via TCP/IP.

Rationale:

- A DNS server provides the capability for a long or complicated TCP/IP location to be accessed by a generic, short alphabetic name. It is basically a lookup service. It maps the generic alphabetic DNS name to its complicated TCP/IP location. The client application programs can be configured to use the generic names when they need to access a database (e.g., a database can be accessed by a client by using the generic name "Summary." The DNS server accepts "Summary" and translates the address into \\UX00001\SRV1\DBASE\DATAWAR.FIL. The client then is able to access the database).

If the database location changes, the DNS configuration is changed, and no changes are needed to each client configuration.
Long binary or text value

Systems Integration - Integration Architecture

Principle 5.00.01 An Integration Architecture enables the inter-operation of multiple technologies into a single integrated network.

Rationale:

- Integration provides a bridge between the hetergeneous operational applications and platforms. An effective architecture ties together the mix of platforms, operating systems, transports, and applications.
- Integration of business applications between agencies and vendors or other agencies supports electronic commerce.

Systems Integration - Integration Architecture

Principle 5.00.02 An Integration Architecture addresses the correlating components of data interchange, business processing issues, and end-user presentation.

Rationale:

- The Integration Architecture encompasses the mulitple layers of new and existing systems and the middleware in between.

Systems Integration - Integration Architecture

Principle 5.00.03 An Integration Architecture meets the needs of linking heterogeous operational application systems while protecting existing investments.

Rationale:

- The Integration Architecture should take into account the need to use existing workstations, peripherals and existing transports to access existing and new applications.

Systems Integration - Integration Architecture

Principle 5.00.04 When making integration decisions, the life span of the solution is a key factor.

Rationale:

- Temporary solution may be engineered very differently than a long-term solution. Cost and effort need to be taken into consideration when providing a solution that is only needed on a temporary basis.
- Short-term solutions are often hard-wired and often have low performance. They are designed to be replaced or easily removed. Cost and effort should also be considered for a short-term solution.
- Long term solutions must be standardized, adaptable, and engineered for high performance.

Systems Integration - Integration Architecture

Principle 5.00.05 Integration Architecture relies on middle service tiers such as interface engines, database gateways, messaging, integration services, and third party tools.

Rationale:

- It is more cost effective and easier to maintain applications that use middle service tiers than to modify multiple legacy applications.
- New N-tier applications still need access to the legacy information stored throughout the enterprise.
- Refer to the Middleware chapter for more information middle service tiers not covered in this chapter.

Systems Integration - Integration Architecture

Principle 5.00.06 Minimize the impact to existing application systems.

Rationale:

- To the extent possible, the Integration Architecture should enable new applications to use existing resources with minimal disruption.
- Where possible, use non-invasive techniques for integration.
- Integration requires good communication infrastructure. If the basic network infrastructure is not in place, a single integrated network of application communication cannot be achieved. (Refer to the Network Architecture chapter.)

Systems Integration - Integration Architecture

Principle 5.00.07 Use statewide technologies whenever possible.

Rationale:

- To the extent possible, use the same technologies in the Integration Architecture that are used in the Statewide Technical Architecture.
- Limit the heterogeneity of the technology used in order to simplify integration and enable migration to future technologies.

Systems Integration - Integration Architecture

Principle 5.00.08 Provide maximum flexibility to integrate heterogeneous systems when enhancing existing end-user functionality through the use of a middle service tier.

Rationale:

- Implement the middle tier with standards whenever possible.

Systems Integration - Integration Architecture

Principle 5.00.09 Use existing integration solutions whenever possible.

Rationale:

- Instead of building a new integration technique from scratch, use an existing vendor solution that answers the specific integration needs of an application system.

Systems Integration - Integration Architecture

Principle 5.00.10 Include centralized security management as part of the Integration Architecture.

Rationale:

- Refer to the Security and Directory Services Architecture chapter.

Systems Integration - Application Communication Middleware

Principle 5.00.11 Using application communication middleware is required in a heterogeneous, distributed environment.

Rationale:

- The tiers of a distributed application, which often run on different hardware and operating systems, must communicate.
- Application communication middleware enables both inter- and intra-application communications.

Systems Integration - Application Communication Middleware

Principle 5.00.12 Using message-oriented middleware changes the fundamental design for building distributed applications.

Rationale:

- Message allows asynchronous processing so applications can continue processing after a message is sent.

Systems Integration - Application Communication Middleware

Principle 5.00.13 Using remote procedure calls (RPCs) offer a good migration strategy.

Rationale:

- RPCs are the easiest transition for mainframe programmers. An RPC is simply a subroutine even though it is running a business rule on the network.
- RPCs are a mature technology. They are already bundled with many operating systems and databases.

Systems Integration - Application Communication Middleware

Principle 5.00.14 Minimize the use of distributed units of work.

Rationale:

- Distributed transaction monitors are becoming less and less a requirement as high speed networks and messaging subsystems are deployed.
- The need for a transaction monitor can be eliminated or reduced by using features of message oriented middleware, combined with application design.

Systems Integration - Application Communication Middleware

Principle 5.00.15 Do not use database middleware for application communication

Rationale:

- Database middleware has limited usefulness. It allows an application component to access data, thereby supporting a two-tier application design.
- Database middleware does not have the capability to provide all levels of inter-component communications. Stretching its use to inappropriate environments will ultimately result in systems that have performance problems.

Systems Integration - Application Communication Middleware

Principle 5.00.16 Using a broker facilitates reuse and shortens development cycles.

Rationale:

- A broker provides access to common services that can be reused and shared, thus reducing development costs. See Componentware Architecture.
- The state can reduce the resources spent on developing and maintaining "islands of applications," which include redundant code. Application developers can focus on new work rather than rework.
- New applications will be combined of new business rules and common shared business rules. Since part of the application is "pre-written" and "pre-tested," delivery of the total application should result more quickly.

Systems Integration - Application Communication Middleware

Principle 5.00.17 Precede selection of application development tools with an application communication middleware strategy.

Rationale:

- In the long term, use of middleware by many applications is of more strategic importance than any one application development tool. Middelware selection should drive the choice of application development tools, not vice versa.
- A range of communication methods is available through middleware. A combination of products may be required.

Systems Integration - Application Communication Middleware

Principle 5.00.18 Select third-party middleware rather than middleware supplied with a development tool.

Rationale:

- De-coupling the middleware from the application development tool provides more flexibility in changing development tools in the future. For example, integrated CASE tools often provide third-party message oriented middleware as well as their own, proprientary message oriented middleware.
- When given the choice of proprietary middleware versus third party middleware, select the third party middleware option. For instance, message oriented middleware provided by the integrated CASE tool vendor limits flexibility and links to a specific vendor and product strategy more closely.
- If message oriented middleware is linked directly to a specific development package, then there is the risk of limited usefulness with other applications that are not developed with the same tool.

Systems Integration - Application Communication Middleware

Principle 5.00.19 Document application programming interfaces (APIs) and interface definition language (IDL).

Rationale:

- APIs and IDL for components and services must be documented so that developers know where they are and how to use them.
- For more information about components and services, refer to the Application and Componentware Architecture chapters.

Systems Integration - Application Communication Middleware Types

Best Practice 5.01.01 When possible, design applications to use asynchronous communication.

Rationale:

- Message oriented middleware supports asynchronous communications.
- Asynchronous messaging requires a distinctly different design. It is implemented with a very basic set of message oriented middleware commands.
- Message oriented middleware provides a reliable form of communication.
- Asynchronous communication offers more flexibility than synchronous communication. The downstream application has more control over its operation.

Systems Integration - Application Communication Middleware Types

Best Practice 5.01.02 Use Remote Procedure Calls (RPCs) when message oriented middleware is not available.

Rationale:

- RPCs provide an acceptable, albeit limited, method of communication between software components.
- RPCs require synchronous communication and are less efficient in the use of resources; they tie up resources from both the client and the server until the service has been provided.
- Synchronous communication requires error handling in the client application if the request is made while a server is unavailable.

Systems Integration - Application Communication Middleware Types

Best Practice 5.01.03 Use distributed transaction monitors only when distributed transactional integrity is required.

Rationale:

- Transaction processing monitors offer significant functionality in addition to transaction management. Using transaction processing monitors when other middleware services suffice may cause unnecessary overhead and may result in performance problems.
- Since distributed transactions are composed of multiple discreet functions, parts of transactions may be at risk for some period of time. Designing applications to eliminate the distributed units of work reduces the risk of transaction failure.
- When distributed transactional integrity is needed, use a TP monitor rather than the transaction management capability of a database management system.

Systems Integration - Application Communication Middleware Brokers

Best Practice 5.02.01 Manage a statewide broker as a strategic infrastructure component.

Rationale:

- The service broker is a critical part of the distributed computing environment because it allows the technical architecture to meet the three goals of efficiency, sharing of information and agency autonomy.
- Strategic infrastructure benefits all agencies and should be centrally managed.

Systems Integration - Application Communication Middleware Brokers

Standard 5.02.01 Use of the service broker is required for inter-application communication.

Rationale:

- The service broker was put in place due to the lack of standards for inter-application communication types such as RPC, MOM, and TP monitors. (See Technical Topic 1,

Application Communication Middleware Types, the Standards section, for more information.)
- While the lack of standards is not an issue for development of any single application, it poses problems for communication between applications. The broker is proposed as a standard communication paradigm for inter-application communication.

Systems Integration - Application Communication Middleware Brokers

Best Practice 5.02.02 Be sure a statewide broker is independent of code development tools.

Rationale:

- The purpose of the service broker is to facilitate communication in a multi-platform, multi-language environment. If the service broker is tied to a single vendor's product, then the goal of facilitating communication in a diverse environment has not been met.
- Implementing a service broker that supports multiple vendors' products helps protect the state from being negatively impacted by market forces.

Systems Integration - Application Communication Middleware Brokers

Best Practice 5.02.03 Be sure a statewide broker provides a suite of communication middleware features.

Rationale:

- A best of breed approach should be taken when selecting the application communication middleware.

Systems Integration - Application Communication Middleware Brokers

Best Practice 5.02.04 Use the state's inter-application middleware, the service broker interface, for inter-application communication between state-developed applications. For interfaces with other applications, use the Interface Engine.

Rationale:

- State-developed applications gain performance and flexibility by using the service broker for inter-application communication.
- In-house or out-sourced custom-developed applications requiring inter-application communication should be capable of using a service broker. Applications sharing or requiring services from external application systems should provide the capability to use the standard inter-application communication middleware architecture.
- In instances where the application code cannot be modified, such as purchased applications where the state does not have rights to source code, use the interface engine. For more information about application integration, refer to the Integration Architecture chapter.
- For more information on the Interface Engine, see the Integration Architecture chapter.

Systems Integration - Application Integration

Standard 5.03.01 Clearly define Application Interfaces.

Rationale:

- To integrate applications for which the state has no source code rights, application interfaces must be clearly defined in order to allow reliable communication between applications.
- To facilitate puchase of best-of-breed software while easing application integration issues, the application interfaces must be clearly defined.

Systems Integration - Application Integration

Best Practice 5.03.01 Anticipate future usage.

Rationale:

- Whenever an application integration is constructed, anticipate future usage so the technology will be adaptable and scaleable.
- For example, an organization may not want to use a screen scraping interface if future requirements are for faster performance or additional information that is not supplied by an existing terminal screen.

Systems Integration - Application Integration

Standard 5.03.02 The message structure must be documented.

Rationale:

- A message or transaction is the mechanism for extracting data from an application or sending data to an application.
- Programmers integrating applications need to know record length and type (ie, whether it is a variable or fixed length record, and if it is variable, the delimiting characters used to separate the fields), and know which fields are optional versus required.
- A description of the data for each field is also necessary.
- Explanations and examples of record formats and field descriptions are helpful and should be included.

Systems Integration - Application Integration

Best Practice 5.03.02 Use application integration strategy for online transaction program (OLTP) application systems, not decision support systems (DSS).

Rationale:

-Data warehouses or other solutions shouuld be used in decision support applications. (For more information on data warehouses, refer to the Information Architecture chapter.)

Systems Integration - Application Integration

Standard 5.03.03 The application must be able to transmit and receive messages using a client/server model.

Rationale:

- The client is the process that sends or originates the message. The server is the process that receives the message.
- Clients and servers may communicate using TCP/IP and sockets, or other communication protocols, such as Serial and FTP, as long as they perform the same transmit and receive functionality.
- Packetization characters, which identify the start and end block strings, and message acknowledgement format must also be provided.

Systems Integration - Application Integration

Best Practice 5.03.03 Design an integration solution that does not write directly to an operational database.

Rationale:

- Existing application logic or busines rules should be used when updating an application database.
- An external user or application could inadvertently corrupt operational data.

Systems Integration - Application Integration

Best Practice 5.03.04 Consider a screen scraping solution when an application link needs to be non-invasive and there are no other non-invasive interfaces available to an application system.

Rationale:

- Using existing terminal screens can be a viable alternative to re-coding a legacy or purchased application for a program interface. Legacy and purchased application screen formats are normally static and contain the information needed by a new application.
- If screen scraping is used through an interface engine, it can be more reliable and stable than a PC screen scraping solution for operational applications.

Systems Integration - Application Integration

Standard 5.03.04 Purchase line-of-business application software rather than custom developing it whenever possible.

Rationale:

- Purchase line-of-business application software can permit the state to respond to business needs in a more timely manner than custom developing software.
- Published APIs are insuffient because their use requires custom development of state applications and it may be impossible to interface two purchased applications. Use of an interface engine provides maximum flexibility.

Systems Integration - Application Integration

Best Practice 5.03.05 Use direct program-to-program intefaces for high transaction volumes.

Rationale:

- Direct program-to-program interfaces pass only the required information between applications, so performance and throughput is at the optimal level.

Systems Integration - Application Integration

Best Practice 5.03.06 When designing an application integration solution using an interface engine, give careful consideration to the design and planning of the application interfaces and connectivity.

Rationale:

- At the beginning of the design stage, involve application developers who are knowledgeable in the business rules and interfaces to each system that needs to be accessed.
- Some application systems may have multiple entry or exit points that can be used. If a non-invasive solution is selected, capitalize on using the entry or exit points that best apply to your application needs.

Systems Integration - Electronic Data Interchange (EDI)

Best Practice 5.04.01 EDI should be fully integrated into the business process and the computer applications that support the process.

Rationale:

- The maximum benefits of EDI can be achieved when EDI is integrated into the business process.
- Focus on the business process supported by EDI rather than on the technology used by EDI.
- Maintain an audit trail that supports tracking and control for EDI transactions. Entries should be maintained for each handoff of a transaction between EDI application elements and between trading partners.
- Where practical, use reciprocal transactions to achieve application level acknowledgement of electronic transactions.

Systems Integration - Electronic Data Interchange (EDI)

Standard 5.04.01 Use ANSI X12 or UN-EDIFACT for Electronic Data Interchange (EDI).

Rationale:

In the United State, most ED transactions comply with ANSI standards approved by the ANSI X12 committee. These standards cover a wide range of commercial interaction, including, but not limited to the following:
- Requition, request for quotation, purchase order, purchase order change.
- Vehicle service order, product service claim, and product service claim response.
- Air freight information, motor carrier bill of lading, U.S. Customs status information, shipping instructions.
- Remittance, credit/debit adjustment, mortgage credit report, real estate title evidence, electronic filing to tax return.
- Student load application, student educational record, student enrollment verification.

Most international EDI transactions comply with the UN-EDIFACT standards endorsed by the United Nations. EDIFACT standards are similar to ANSI X12 standards. EDIFACT standards cover EDI for administration, commerce, and transport. Note the ANSI s committed to migrating the X12 standards to be compatible with UN-EDIFACT standards.

Systems Integration - Electronic Data Interchange (EDI)

Best Practice 5.04.02 Use EDI to automate frequently used business transactions.

Rationale:

- EDI is cheaper, faster, and more accurate than performing the same transactions manually.
- EDI is appropriate for agency-to-agency transactions should exchange mapped data, without requiring the trading partners to go through the generation/interpretation process. This avoids the requirements that both internal trading partners buy and maintain EDI software.
- The state may require EDI.

Systems Integration - Electronic Data Interchange (EDI)

Best Practice 5.04.03 Use a single EDI software package for the entire enterprise, even if there are mulitple EDI servers.

Rationale:

- A single best-of-breed EDI software package reduces the complexity of implementing and managing EDI across the state.
- Business process and transaction volume may require additional installation of EDI software, but all should be the same EDI package.

Systems Integration - Electronic Data Interchange (EDI)

Best Practice 5.04.04 Use industry standard for transactions that are being performed electronically.

Rationale:

- For commerce between agencies, or between agencies and US-based trading partners, use the latest version of the ANSI X12 transaction set.
- Be aware that standards will evolve over time, and all participating trading partners will need to synchronize when a newer version is implemented.
- If there are no standards for transactions conducted frequently, define some with the major trading partners.

Systems Integration - Electronic Data Interchange (EDI)

Best Practice 5.04.05 Manage EDI as a critical application.

Rationale:

- EDI has a significant impact on the state's business. Even when EDI software is deployed on a PC-based server, it must be managed as a mission-critical application.
- Successful EDI implementations require transaction volume and capacity planning. (Refer to the System Management Architecture chapter for more information about capacity planning.)
- Secure EDI transactions with a level of security appropriate for the business function being performed.

Systems Integration - Electronic Data Interchange (EDI)

Best Practice 5.04.06 Use a value added network (VAN) for data transmission to outside trading partners.

Rationale:

- Direct file transfer (eg, "batch feeds") is acceptable for EDI performed within an agency or between agencies.
- Direct EDI introduces additional complexity into the transmission and receipt of transactions to outside trading partners.
- Look for opportuntity to use the Internet for EDI data transmission when standard software is available to handle Internet EDI.
- When performing financial EDI, also use an automated clearinghouse to manage the funds transfer portion of finanancial transactions.

Systems Integration - Electronic Data Interchange (EDI)

Best Practice 5.04.07 Purchase - do not build - software to perform EDI translation, formatting, and transmission to a VAN.

Rationale:

- Start small with a single transaction set to a single key trading partner; add additional trading partners and transactions after the others are running smoothly.

Systems Integration - Electronic Data Interchange (EDI)

Best Practice 5.04.08 Use an EDI solution over an interface engine solution if possible.

Rationale:

- For EDI transaction sets, the EDI software has built-in capability to format transaction data.
- When using EDI, only one interface program is need: the originating application (for outgoing transactions) or the target application (for incoming transactions).

Systems Integration - Electronic Data Interchange (EDI)

Best Practice 5.04.09 If EDI software does not include data mapping capability, use an interface engine, rather than custom programming for data mapping.

Rationale:

- Data mapping using an interface engine is noninvasive.

Systems Integration - Data Access Integration

Best Practice 5.05.01 Use as few middleware layers as possible when implementing a database gateway.

Rationale:

- Additional layers of middleware in between an application and the database gateway could hinder performance of mission critical applications. For example, an application that needs to access a database gateway can implement an ODBC middleware layer that ultimately accesses the gateway middleware. Application performance can be increased if the application was written to make direct calls to the gateway middleware, omitting the ODBC layer.
- If there are fewer middle conversion tiers, there are less operational layers to maintain in the event of maintenance or upgrades. For example, if there is a change to an application database location, or an upgrade or maintenance update to the middleware software, it can effect all end user workstations and servers that access that application.

Systems Integration - Data Access Integration

Standard 5.05.01 There is no Remote Procedure Call (RPC) standard. Use the state of North Carolina's service broker for inter-application communication.

Rationale:

- Even with an RPC that is endorsed by a vendor neutral party, such as the Open Group, there is no standard RPC.
- RPCs are available from different vendors, such as the Open Group's DCE RPC, Sun Microsystem's ONC/RPC, and Microsoft's RPC.
- Each vendors version has a different application programming interface and they do not inter-operate with one another.

Systems Integration - Data Access Integration

Best Practice 5.05.02 Balance the type of data access method implemented with required performance needed by the application end users and the impact to the existing operational databases.

Rationale:

- If the wrong data access method is selected, the performance may not match the application needs.
- A solution that is good for a new application may adversely impact existing operational applications.

Systems Integration - Data Access Integration

Standard 5.05.02 There is no Message Oriented Middleware (MOM) standard. Use the state of North Carolina's service broker for inter-application communication.

Rationale:

- At present, all message oriented middleware is proprietary. Products from different vendors have different application programming interfaces, which do not inter-operate with one another.

Systems Integration - Data Access Integration

Best Practice 5.05.03 Keep the integration strategy as simple as possible.

Rationale:

- The more complicated the strategy, the more difficult it is to maintain and change.

Systems Integration - Data Access Integration

Standard 5.05.03 There is no distributed transaction processing (TP) monitor standard. Use the state of North Carolina's service broker for inter-application communication.

Rationale:

- The applications coordinated by a tranacation monitor which will run on different platforms with access to different databases and resource managers.
- The applications are often developed using different tools and have no knowledge of one another.
- Industry standards specify how a TP monitor interfaces to resource managers, other TP monitors, and its clients.
- X/Open XA specification defines specfications for two-phase commits that work with distributed databases.
- X/Open TX standard defines transactions.
- X/Open X/ATMI provides a standard transaction management interface.

Systems Integration - Data Access Integration

Best Practice 5.05.04 Code data integrity verification rules into the DBMS whenever possible, particularly when external users and programs will be writing data directly to the DBMS.

Rationale:

- Since most DBMS vendors can code triggers and rules into the database, it is recommended to use this technology wherever possible in order to ensure data integrity.
- For more information on databases, refer to the Data Architecture chapter.

Systems Integration - Data Access Integration

Best Practice 5.05.05 Separate decision support systems (DSS) from online transaction processing (OLTP) database whenever possible.

Rationale:

- If this practice is feasible, it will reduce the impact of ad hoc and large queries from decision support systems onto production operational application databases that are used by online users for day-to-day operations.

- For more information on database architecture, refere to the Data Architecture chapter.

Systems Integration - Terminal Integration

Best Practice 5.06.01 Design new application systems with the user interface as a separate application tier.

Rationale:

- If a GUI is not designed as a separate tier, and a character-based terminal interface to the client/server application is not possible, terminal integration will not be successful.
- When building a client/server application that needs to be accessed by a wide variety of end users and platforms, ensure that it can be accessed by any type of user interface, including graphical user interface and character-base interfaces.

Systems Integration - Terminal Integration

Best Practice 5.06.02 Base the user interface design on the targeted end user platform.

Rationale:

- If the inteface is similar to existing user interfaces, the learning curve is reduced and the end users can easily become productive on new systems.
- If a GUI-based interface is used, conform to established GUI-based standards.
- If a 3270 terminal interface is used, design the interface similar to existing screen interfaces that are used by legacy systems.

Systems Integration - Terminal Integration

Best Practice 5.06.03 Implement as few comminications tiers are possible.

Rationale:

- With fewer tiers the architecture will be simpler and easier to maintain.

GroupWare - Groupware Architecture

Principle 6.00.01 Groupware requires a consistent infrastructure to truly support collaboration and communication.

Rationale:

- Content exchange, directory services, and authentication services are key infrastructure components necessary to facilitate communication and collaboration.
- Email is the communication infrastructure component within and outside the state. Today, a significant portion of the business communications occurs across email services

GroupWare - Content Exchange

Best Practice 6.01.01 Avoid proprietary formats in anticipation of document exchange with outside users and applications.

Rationale:

- Proprietary formats inhibit document exchange, and may create future barriers to communication.
- If proprietary formats are used, the capability must be provided to convert documents to standard formats for content exchange. Any vendor using proprietary formats should provide a conversion routine.

GroupWare - Content Exchange

Standard 6.01.01 For non-editable documents, the standard file format is PDF. Typical application software using this file format includes word processing, imaging systems, and World Wide Web publishing.

Rationale:

- PDF is widely deployed for non-editable content exchange.
- The reader is available at no cost

GroupWare - Content Exchange

Standard 6.01.02 For monochrome documents or drawing, the standard file format is TIFF using CCITT/ITU Group IV Compression.

Rationale:

-Typical application software using this file format include: word processing, archive and retrieving, workflow, mulitmedia, medical systems, digital publishing, pattern recognition, and geographic information systems.

GroupWare - Content Exchange

Standard 6.01.03 For color documents, drawings, or photographs, the standard file formats are GIF and JPEg.

Rationale:

- Typical application software using this file format includes multimedia, work processing, medical systems, digital publishing, and geographic information systems.

GroupWare - Content Exchange

Standard 6.01.04 For facsimile documents, the standard file format is TIFF using CCITT/ITU Group III compression.

Rationale:

- Typical application software using this file format includes work processing, archival and retrieval, and workflow.

GroupWare - Content Exchange

Standard 6.01.05 - For vector or geometric data, the standard file formats are DGN and DWG.

Rationale:

- Typcal application software using this file format includes CADD and geographic information systems.

GroupWare - Content Exchange

Standard 6.01.06 For multiple images, the standard file format is MPEG-1.

Rationale:

- Typical application software using this file format is multimedia.

GroupWare - Electronic Mail

Best Practice 6.02.01 Email servers should be administered and managed as a part of the strategic infrastructure.

Rationale:

- A properly structured email system can provide the state with a comprehensive, effective, inexpensive, and widespread method of communication, while permitting a choice of email clients.
- Email is a valuable tool because it provides the structure for easily moving messages and attachments in a timely manner between clients, thereby increasing workflow and productivity.
- Email service should be available at all times from any location. Time, distance, and location should not restrict email service.

GroupWare - Electronic Mail

Standard 6.02.01 Use Simple Mail Transport Protocol (SMTP).

Rationale:

- Simple Mail Transport Protocol (SMTP) is the standard transport protocol for sending messages from one MTA to another MTA over the Internet. Using MIME encoding, it enables the transfer of text, video, multimedia, images, and audio attachments. It is the predominate transfer protocol utilized by web browser-based email user agents.

GroupWare - Electronic Mail

Best Practice 6.02.02 Email servers should support multiple email clients.

Rationale:

- A properly structured email system can provide the state with a comprehensive, effective, inexpensive, and widespread method of communication, while permitting a choice of email clients.

GroupWare - Electronic Mail

Standard 6.02.02 Use Multi-purpose Intenet Mail Extensions (MIME)

Rationale:

- Multi-purpose Internet Mail Extensions (MIME), a SMTP message structure, is the standard specification for the attachment of audio, video, image, application programs, and ASCII text messages. The content type is stored in the message header as mail extensions. When the message is delivered, the player or application specific to the content type is opened so that the attachment can be viewed in its native format. If the player or application is not included with the browser, then the user must load it. Common image and video players are included with most browsers.
- The MIME standard will require standardization of certain protocols in the near future. By its definition, MIME is transformable. Although two applications may be MIME-compliant, each application can use a proprietary or custom set of extensions. The data associated with the proprietary extensions may be lost in transfer. Common protocols cut down on the risk of a loss of data occurring.

GroupWare - Electronic Mail

Standard 6.02.03 Use Internet Message Access Protocol version 4 (IMAP4).

Rationale:

- Internet Message Access Protocol version 4 (IMAP4) is the standard protocol for access to the mail server. The user has the option of storing and manipulating messages on the mail server, which is important for job functions that require the user to access mail from several different clients. IMAP is also ideal for situations where the user has a low speek link to the mail server. Instead of downloading all messages to the client, IMAP allows the user to select which specific messages to download. If a message has several MIME attachments, the user can specify that only the text portion of the message is to be downloaded for viewing. This practice is considerably more efficient in the event that a high-speed link is not readily available.
- Note: Options sometimes exist to configure mail servers and clients without IMAP4 settings. Email servers and clients should be implemented using IMAP4.

GroupWare - Electronic Mail

Best Practice 6.02.03 Select a C&S application that allows the user to create both public and private notification groups and contact lists.

Rationale:

- Public and private notification groups are lists of people and/or resources that have common calendar and schedule needs, such as project groups or a list of conference rooms. Users can assign selected individuals to their private groups, and administrators can assign selected individuals to public groups. When a group name is entered as a participant, available times can be selected based on the group's information, and a notification message is forwarded to all individuals included in that group. This feature streamlines the use of C&S applications, making them more efficient.

GroupWare - Electronic Mail

Best Practice 6.02.03 Use a common email directory service throughout the state.

Rationale:

- An enterprise-wide email directory service should be accessible by everyone within the organization. The statewide directory service should be a seamless integration of each agency's directory service. If a user in one agency requests an email address for a user in another agency, the action should be transparent, without the requester knowing where in the organization the address is stored.
- The directory service should be compatible with the directory services of other components in the network. Other applications require use of an email directory service. Use of a single directory service will facilitate reuse of information and directory access routines. It is necessary for heterogeneous components to access the directory service.
- For more information about directory services, refer to the Directory Service sub-topic in this chapter.

GroupWare - Electronic Mail

Best Practice 6.02.04 Select a C&S application that enables task and resource management.

Rationale:

- In addition to people, the user must have the ability to schedule facilities and equipment. For example, the user should be able to reserve a meeting room and an overhead projector through the C&S Application.
- The C&S application should be capable of tracking tasks (e.g. "To Do" lists that automatically send reminder messages for upcoming deliverables

GroupWare - Electronic Mail

Standard 6.02.04 Use Lightweight Directory Access Protocol (LDAP).

Rationale:

- Lightweight Directory Access Protocol (LDAP) is the standard directory access protocol. LDAP is based on Directory Access Protocol (DAP), and X.500 standard access protocol. X.500 is a set of CCITT/ITU standards for electronic directory services. LDAP has been proven to be more efficient for MUA to directory services transactions. In addition, LDAP can be utilized to access databases other than the email directories, which will add value to other groupware applications, such as scheduling.

GroupWare - Electronic Mail

Standard 6.02.04 Plan for adaptability to accommodate future changes in the email environment.

Rationale:

- Protocols are constantly evolving. It is best to use applications that easily adapt to changes in the environment without requiring a high degree of custom or proprietary code. Applications should be capable of supporting multiple protocols and services.
- New technology is being rapidly developed and perfected. In the near future, the marriage of email and telephony will be complete. It is prudent to purchase applications today that can readily be re-configured to accept the technologies of tomorrow. The email system should grow stronger with future changes, not slip into obsolescence.

GroupWare - Electronic Mail

Standard 6.02.05 Select an email server system that allows multiple standards-based email clients.

Rationale:

- When an email server uses IMAP4 standard, any IMAP4-base client can access that server.

GroupWare - Electronic Mail

Standard 6.02.05 Select an email client that includes standard APIs for email-enabling other applications.

Rationale:

- Email is a key component of workflow. If a user is working on a document and chooses to send that document to another user, the user should not have to close the document creation application and open email to send it. Instead, the user should be able to mail the document directly from the native application.
- Calendaring and scheduling applications can use mail message delivery for meeting proposals.
- Common APIs in use today are the messaging application programming interface (MAPI) , vendor independent messaging (VIM), and common messaging calls (CMC).

GroupWare - Electronic Mail

Best Practice 6.02.05 Select a C7S application that allows remote and proxy access.

Rationale:

- The user should be capable of disconnecting from the network and still maintain access to personal and shared calendars. Any updates to the calendars or schedule notifications made while the user is offline should be uploaded and synchronized at the time that the user reconnects to the network.
- Incoming schedule notifications should be held by the C&S server until the user reconnects to the network at which time the messages are synchronized with the user's local calendar.
- Proxy access enable a C&S user to allow another authorized user or users to administer their personal calendar. This function may be limited to viewing, or allow full maintenace capabilities.

GroupWare - Electronic Mail

Best Practice 6.02.06 Select a C&S application that can be accessed through a web front-end.

Rationale:

- Both Intranets and Internets are accessible by anyone within the organization via a web-browser. Web enabled C&S applications allow users access to C&S information for anyone, anywhere in the state.

GroupWare - Electronic Mail

Standard 6.02.06 Implement security for email message transport and storage

Rationale:

- Private and official correspondence will require varying degrees of protection including authentication and encryption. SMTP/MIME was created as a means of "casual" communication over the Internet. It was not created to be a completely secure medium. Protocols are currently being developed bridge the security gap.

GroupWare - Calendar and Scheduling

Best Practice 6.03.01 Select an open C&S application, which maintains transparent interoperability with other C&S applications and computing platforms used across the state.

Rationale:

- The C&S system must be capable of supporting multiple server platforms and client platforms. The operating environment within the state is, and will remain, heterogeneous. The C&S system must therefore be capable of transparently transferring schedules and meeting information across each of the operating systems.
- C&S applications are typically purchased independently by each agency based on the particular needs that the agencies must satisfy. The selected applications must be capable of exchanging schedules, notifications, and material with the C&S applications utilized by other agencies. The use of one application must be capable of viewing a user's calendar created and stored on another application.

GroupWare - Calendar and Scheduling

Standard 6.03.01 There are currently no standards pertaining to this technical topic.

Rationale:

- There are cuurently no standards pertaining to this technical topic.

GroupWare - Calendar and Scheduling

Best Practice 6.03.02 Select a C&S application that provides a mechanism for attaching supporting documentation, such as meeting materials, to the notification message.

Rationale:

- The capability to include or attach meeting agendas, supporting documentation, and deliverables maximizes the efficiency of C&S as a productivity tool.

- The user should not be required to compose a separate email message to send attachments. The transport mechanism for attachments should be accessible from the scheduling application.

GroupWare - Document Management

Standard 6.04.01 Conform to NC Government Information Locator Service (NC GILS).

Rationale:

- Where applicable EDM applications, databases, and repositories, should be designed to interface with and conform to North Carolina Government Information Locator Service (NC GILS) standards being promulgated by the NC Office of State Planning (OSP).
- The U.S. government created a government information locator service (NC GILS) to assist in locating information from federal, state, or local government agencies. This service is an electronic "card catalog," describing and indexing information available from federal agencies. By following internationally recognized standards, (the information search and retrieval service and protocol standard, ANSI/NISO (Z39.50), the information supplied by the disparate agencies includes the same content items and is available for searching electronically with information retrieval software via the Internet. The GILS standard has been adopted by several states and foreign countries, which makes it the foundation of a global information locator service.
- As a result of Exective Order 100 from Governor Hunt, North Carolina is developing a government information locator service compliant with the federal GILS. North Carolina's implementation of GILS is called NC GILS.
- State and local agencies in North Carolina are required by the Public Records Law (North Carolina G.S. 132-6.1 (b)) to index all databases created or significantly modified after a certain date. NC GILS provides a means to output the indexing information in a standard way to allow searching across agencies and across other state, local, and federal agencies and political jurisdictions. Thus, NC GILS will link public database indexes created in response to the amended Public Records Law to indexes developed by the federal government, other states, and other counties, to provide for the global exchange of public information. OSP is currently working with State Public Records Cataloging Services to determine indexing standards as they relate to documents.
- For more information about NC GILS, refer to the web site: http//www.ncgils.state.nc.us/NCGILS.

GroupWare - Document Management

Best Practice 6.04.01 Evaluate potential requirements over a longer term basis and implement a "platform" that can be used to develop document-enabled applications and provide a uniform approach to document storage and access.

Rationale:

- Use the standards guidelines to assure the implementation of "scalable," open systems.

GroupWare - Document Management

Standard 6.04.02 Conform to NC Public Records Law.

Rationale:

- The development of EDM systems, management of public records, and explosion of e-mail means that system design efforts and policy and procedure as to the processing, routing, retention, and disposition of data and documents must be accomplished with respect to public law.
- Any database that falls under the Public Records Law (North Carolina 132-6.1 (b) and G.S. 121) must be included in the State Public Records Cataloging Services (SPRCS) under the development of Cultural Resources (DCR).
- SPRCS was developed by the Division of Archives and History as a series of guidelines, catalog services, procedures and data used by the SPRCS work unit in managing North Carolina's public records.
- For more information about SPRCS, refer to the web site: http//www.spr.dcr.state.nc.us

GroupWare - Document Management

Best Practice 6.04.02 Assure the availability of open application program interfaces.

Rationale:

- Adhere to APIs and integration standards being advanced by the Association for Information and Image Management.

GroupWare - Document Management

Standard 6.04.03 Implement document management systems and components that conform to the Document Management Alliance specifications (DMA 1.0 and ODMA 2.0).

Rationale:

- There are numerous issues related to interoperability among document management applications, services, and repositories. Standards are needed to manage the increased life expectancy and complexity of re-usable electronic documents and content.
- The Document Management Alliance (DMA), is a task force of AIIM. DMA conforming products will supports open design for user interfaces, workstations, network operating systems and services. The DMS provides a framework for vendors to deliver products that provide query services (simple transparent access from every desktop to information anywhere on the network), and library services (including check-in and checkout, version control, security, and accountability). The DMA is working with the Open Document Management API (ODMA) group which specifies the common application program interfaces, and high level call interfaces that enable other client applications (such as MS Office) to work seamlessly with DMA compliant document management systems.
- For more information about AIIM standards programs, refer to the web site: http//www.aiim.org/industry/standards.

GroupWare - Document Management

Standard 6.04.03 Execute a "pilot" project based on an identified business need and defined business process rather than just a pilot of technology.

Rationale:

- A pilot project will provide an early demonstration of success and allow an organization to begin to absorb and understand all the business cultural isssues associated with the concepts of EDM, workflow, and knowledge management.
- These concepts are changing the traditional ways of thinking about business applications. Business processes will most likely need some level of redesign.
- For imaging applications, many organizations start with records management requirements; the need to create a permanent record of paper files that would otherwise be boxed and sent to archives. The existiing business workflow will require changes introduced by technology. Even a simple records application may require 1) scan, 2) quality control, 3) re-scan, 4) index, 5) verify, 6) commit to permanent storage.

GroupWare - Document Management

Standard 6.04.04 Implement workflow systems that conform to the interface specifications of the Workflow Management Coalition (WfMC).

Rationale:

- WfMC is another working group of AIIM and is closely aligned with the work of the DMA. As automated workflow systems continue to evolve, the complexities associated with a common approach to process definition, process repositiories, object manipulation and transport, and user interfaces are enormous. The Workflow Management Coalition (WfMC) has proposed a framework for the establishment of workflow standards. This framework includes five categories of interoperability and communication standards that will allow multiple workflow products to coexist and inter-operate within a network environment. This framework is contained within a Reference Model for workflow management systems that includes five interface specifications. The model includes the following:
- Process Definition Tool.
- Workflow Enactment Services.
- Workflow Client Applications.
- Invocation of Native Applications.
- Workflow Package Interoperability
- At this time, there are many companies designing products that comply with one or more of these interface specifications. Agencies plannning production workflow applications that need to route work outside of the production system for processing or decision making should work carefully with vendors and service providers to determine functional requirements and WfMC standards compliance.
- For more information about the WfMC and the work of the coalition refer to the Web site at: http:/www.aiim.org/wfmc/index.html.

GroupWare - Document Management

Best Practice 6.04.04 Define business processes as conversational procedures, not according to information/data flow.

Rationale:

- If the process cannot be described step by step language, it cannot benefit from automation.
- Specify document handling and workflow as a series of conversations.
- Use structured methods.
- Draw simple pictures.
- Use a conventional template.
- Never model the information flow.

GroupWare - Document Management

Standard 6.04.05 Use Adobe Acrobat Portable Document Format (PDF) for Non-editable Electronic Documents.

Rationale:

All documents in final form are prepared for distribution and publishing with no intention for further modification must be stored and delivered in Adobe PDF format.
- For more information about PDF refer to the web sites:
- http://www.state.nc.us/adobe/irmc1.htm
- http://www.abobe.com/prodindex/acrobat/adobepdf.html

GroupWare - Document Management

Best Practice 6.04.05 Be prepared to modify processes based on lessons learned from EDMS and workflow applications.

Rationale:

- With this technology, business processes will usually change by collapsing a sequence of steps into a smaller number of parallel steps.

GroupWare - Document Management

Standard 6.04.06 Ensure hardware/software and image file compatibility using TWAIN, ISIS, and TIFF Standards.

Rationale:

- For typical business document imaging applications, software that controls the operation of the scanner (and some other recognition peripherals) is provided. Not all scanner hardware and scan software are compatible. The industry standards to adhere to are TWAIN and more recently ISIS (Image and Scanner Interface Specification).
- The scanned images of typical business documents should be committed to storage in Tagged Image File Format (TIFF) Version 6.0 using CCITT/ITU Group III or IV compression. Organizations planning imaging applications should investigate and demonstrate that any product selected is capable of exporting images in a format that they can be reused. Images that can not be shared are a wasted investment and could result in the loss of critical data.

- Avoid new deployment or migrate away from proprietary image file formats. The current technology direction for image file format is TWAIN. The emerging technology file format is ISIS.

GroupWare - Document Management

Best Practice 6.04.06 Avoid co-mingling manual and automated processes or design clear boandaries and assure accountability for the correctness and completeness of manual steps.

Rationale:

The goal of EDM is to enable the complete automation of day-to-day processes and combine legacy systems with the paperwork (probably images, may be other types of e-docs) that is linked to them. These processes must first be identified and documented to provide a means of analysis.
- Some documents are not suitable for being processed electronically. The feasibility for each document type to be processed electronically must be determined.
- The business rules and procedures must be able to allow processing the majority of a workflow electronically. Switching from electronic to manual processing within a business process will be efficient and leave a larger margin of error.

GroupWare - Document Management

Standard 6.04.07 Select magnetic storage subsystems that adhere to state convenience contract specifications. Select optical storage subsystems based on smaller standard form factors.

Rationale:

- Typical electronic documents, created with office automation suites, will reside on industry standard magnetic disk that is server or network attached. This will generally be transparent to the users of the EDMS. The images of scanned paper documents might also be stored on standard network attached magnetic disk. Magnetic storage will always provide the most performance in the speed of retrieval, and magnetic disk is increasing cost competitive with optical disk storage. When selecting any magnetic storage solution, adhere to other parts of the STA that provide the standards for these types of systems.
- Very large document collections (usually image applications) will probably require optical storage subsystems (many are proprietary) Where there is a requirement for the permanent storage of unalterable documents, optical is chosen in the form of Write Once Read Many (WORM) disks. These types of systems generally involve special software that is used to manage the storage and movement of documents from optical to magnetic when documents are requested by users. Optical disks may be mounted in single standalone drive units or they may be loaded into various sizes of "juke boxes." Software handles the retrieval and loading of specific disks in response to user requests. Typical EDMS systems today will use a 51/4 form factor and will be WORM or Compact Disk type formats. Larger disks are available for specialized applications and are generally proprietary.

GroupWare - Document Management

Best Practice 6.04.07 Match the selected tools to the mission requirements.

Rationale:

Select application components that provide production or transaction-oriented capabilities and ad-hoc capabilities, or recognize the differences between systems/products that are designed for one or the other. Also, consider the scale of the requirements. The components required for the administration of a small office are very different than those needed for the administration of the state's retirement programs.
- Process oriented workflows automate definable, repetitive and well-understood programs and procedures. The automation of these processes provide efficient flow of events and automated management of the process.
- Ad hoc workflows accommodate short lived, unstructured processes. These flows provide flexibility in the work process.

GroupWare - Document Management

Standard 6.04.08 Use eXtensible Markup Language (XML 1.0) when capturing or authoring document content that requires further automated processing by other information systems and web based clients using standard XML enabled browsers.

Rationale:

- This standard is promulgated by the World Wde Web Consortium (W3C).
- XML is a subset of the Standard Generalized Markup Language (SGML, and ISO standard).
- XML encodes a description of a document's storage layout and logical structure with a document type definition (DTD). It provides a mechanism to combine structured data and unstructured information content.
- XML allows information systems and applications to automatically process XML documents when the systems are combined with an XML processor.
- The specification (DTD) describes the required behavior of XML processors in terms of how they read XML documents, and what information they must provide to the processing application. For more information about W3C and XML refer to the Web site: http://w3.org
- The entire contents of the XML 1.0 specification are at: http://www.w3.org/TR/1998/REC-xml-19980210.

GroupWare - Document Management

Best Practice 6.04.08 Select EDM and workflow tools that comply with AIIM open standards, are platform independent, and can be shown to be inter-operable with similar tools and other components of the statewide technical architecture.

Rationale:

- Selecting application components that adhere to industry standards is important for flexibility and adaptability.
- Products must support mulitple server, workstation, and application platforms.

- Workflow components should trigger or send some message-based notification when human intervention is needed (using middleware, email, etc) Partial automation of the office environment will cause confusion as to which activities need human initiative and which activities are being managed by the workflow system. This confusion will cause inefficiencies and leave a wider margin of error in work to be performed.
- Workflow systems should provide a standard interface to other workflow systems for the purpose of passing and processing work items between business units and processes.

GroupWare - Document Management

Best Practice 6.04.09 Select tools that enable reporting of production statistics, real time monitoring of work-in-process, and that reporting for longer-term process performance metrics.

Rationale:

- Tools should collect and report metrics.
- Standard or customizable reports should provide detailed information about documents and work items for the duration of their existence.
- With process automation, identification of inefficiences and bottlenecks can be enabled. While automation speeds up the process, it can also cause too much work to be distributed to a number of people that are unable to process that work effieciently. EDM and workflow should be able to identify problems and thus provide management with the opportunity to balance and modify workflow as needed. The balancing of work queues and prioritization of work distribution can occur. This can ensure that no staff member is left without work while another is backlogged.

GroupWare - Document Management

Best Practice 6.04.10 Determine the various levels of access privileges that users will require, and select products that are tightly integrated to operating system access and security parameters.

Rationale:

- The public should have view-only access to documents. However, public users need to be provided with a mechanism to automatically generate inquires and requests for service.
- In order for the EDMS to improve document collaboration, a document owner must be capable of granting another user permission to edit a document. The EDMS should automatically track and report all events related to the manipulation and flow of documents.

GroupWare - Document Management

Best Practice 6.04.11 As they pertain to electronic document management, adhere to the Application and Data Architecture guidelines in the statewide technical architecture.

Rationale:

- The development process for document enabled applications is not that different than for traditional database applications. EDM systems and components simply provide a more specialized environment for working with information in formats different from tradional hardcopy and computer displays. Existing and emerging products are very compatible with the state's n-tier architecture.

GroupWare - Document Management

Best Practice 6.04.12 Select workstations, servers, and peripherals with the specific needs of different types of users in mind.

Rationale:

- Typical workstations used for office automation suites may not be sufficient for many types of e-docs.
- For imaging applications, take into account technical requirements of data compression, quality assurance, and dual page viewing. Generally, imaging applications will require higher performance for servers, to network, to various types of specialized workstations (eg, scanning, viewing, faxing, and printing).

GroupWare - Document Management

Best Practice 6.04.13 A cost-effective strategy for indexing business documents requires development of a comprehensive indexing scheme containing standard naming conventions.

Rationale:

- The indexing scheme should contain the minimum standard items as required by the State Public Records Cataloging Service and the Office of State Planning.
- The scheme should apply to all types of documents in the organizatoin regardless of their physical format.

GroupWare - Document Management

Best Practice 6.04.14 Consider the need, cost, and resources associated with the conversion of records backfiles, either on paper or on microfilm/fiche.

Rationale:

- Large collections may cost more to convert than the EDM/Imaging system itself.
- A more cost-effective approach for paper archives would be to provide an electronic index to the paper only, and use traditional disposition and permanent storage methods.

Platform - Platform Architecture

Principle 7.00.01 Design servers with bias toward granularity in physical servers.

Rationale:

-Using multiple servers from the same vendor with the same operating system release is cost effective because a group of uniform servers is easier to manage and integrate across a wide geographic area and multiple agencies.

- A highly granular, loosely-coupled server design supports modular application code sets in an N-Tiered application architecture.

Platform - Platform Architecture

Principle 7.00.02 Design mission critical systems without a single point of failure.

Rationale:

- Distributed systems can be designed to be extremely robust.
- Small granular servers make it easier to replicate services for increased availability.
- Systems should be designed to permit continued operations, albeit at reduced throughput, when a server fails in normal operations or in the event of a disaster

Platform - Platform Architecture

Principle 7.00.03 Design all servers implementing a particular application, application suite, or tier within an application with binary compatibility.

Rationale:

-With binary compatibility, there would be no need to recompile an application for different platforms. For example, if an application that is going to be deployed on servers located in employment security offices all servers running that application should be binary compatible -- this must be ensured even if the platforms are from the same manufacturer. The platforms must run the same version of the operating system and must not require any recompilation of the line of business application to deploy from one office to another.
- Total binary compatibility will support automated software distribution across servers and associated strategies which reduce support costs and provide stable computing platforms that can be reliably shared acrossed agencies.

Platform - Platform Architecture

Principle 7.00.04 Utilize open, vendor-neutral systems standards, whenever possible.

Rationale:

- Open, vender-neutral system standards provide flexibility and consistency that will allow agencies to respond more quickly in an environment of changing business requirements.
- Vendor-neutral systems support economic and implementation flexibility.
- Vendor-neutral systems also protect the state against unexpected changes in vendor strategies and capabilities

Platform - Platform Architecture

Principle 7.00.05 Design servers to allow multi-tasking and multi-threading.

Rationale:

- Multi-tasking achieves better CPU utilization.

- Multi-threaded processing enable a server to respond to multiple user requests more efficiently.
- These features also facilitate session management. Fewer sessions to manage provides a more scaleable solution. Multi-threading usually provides capability to execute more sessions ie, more users can run the same application simultaneously, or several threads of the same application can run simultaneously. The ability to run more session or threads would demonstrate a more scaleable solution.

Platform - Platform Architecture

Principle 7.00.06 Design servers to be field upgradeable.

Rationale:

- Rapid changes in business processes are enabled in part by implementing a platform technical infrastructure that exceeds the immediate application requirements. This means agencies should purchase servers with larger chassis so they are able to be expanded more easily and cost effectively.
- Field upgradeable servers provide maximum flexibility and adaptability for growth and new functionality.

Platform - Platform Architecture

Principle 7.00.07 Make platform decisions based on long-term business needs.

Rationale:

- Picking a platform to run a specific purchased application to meet a specific business need may not support long-term state and local agency requirements. In addition, investment in short-term solutions that do not migrate easily to a strategic platform can ultimately be more costly and time consuming than investing in a strategic platform initially.
- Cooperative efforts among agencies and local government to cross-utilize platform incorporating the state's recommended platform technology components may provide the most cost-effective solution. File and print servers, for example, can easily be implemented as shared resources.

Platform - Server Platform

Best Practice 7.01.01 Run mid-range application and database servers on a 32-bit multi-tasking, multi-threaded operating system, at a minimum.

Rationale:

- Migration from 16-bit operating system platforms to 32- or 64-bit operating system platforms will support faster processing, access to more memory, and better memory and process management.
- In an N-tiered client/server environment, speed, memory capacity, and memory and process management become increasing important as processing is distributed across platforms.
- The 32- and 64-bit operating systems provide more stable, reliable platforms in an N-tiered, distributed client/server environment.

Platform - Server Platform

Standard 7.01.01 Run Distributed application server on platforms supporting "open" operating systems.

Rationale:

- Open operating systems are available from mulitple vendors, such as Unix
- Open operating systems run on hardware available from multiple vendors, such as Windows NT.
- Open operating systems are in the public domain, but have significant industry support, such as Linux.

Platform - Server Platform

Best Practice 7.01.02 For reliability and ease of support, place each major application on a uniformly configured server. This may require that each major application be implemented on its own server.

Rationale:

- Use the same reference configuration of these servers. Important items to consider when plannng for consistency include using the same versions of network software, using the same network hardware cards, etc.
- Tuning performance through configuration changes can make overall maintenance more difficult. In the long run, it may be less expensive to buy more powerful hardware than it is to spend time on individualized tuning and maintenance.
- The network Operating System should be considered a major application and run on its own platform.

Platform - Server Platform

Standard 7.01.02 Make sure server platforms are POSIX compliant.

Rationale:

- POSIX is an IEEE standard design to facilitate application portability and interoperability. This facilitates movement of applications from one platform to another if needed.

Platform - Server Platform

Standard 7.01.03 Make sure server platforms comply with third party certifications.

Rationale:

- Third party certfications foster quality product purchases from manufacturers that have demonstrated abilities to deliver and support these products.

Platform - Server Platform

Best Practice 7.01.03 Consider normal anticipated future application growth when determining capacity requirements for server platforms.

Rationale:

- A server platform should be purchased that will accommodate the current demand as well as support anticipated normal growth without requiring the purchase of a new server chassis.
- Rather than purchase a fully configured server, purchase the next larger size platform to allow for expansion. This will permit upgrades to an existing platform to accommodate growth rather than forcing the purchase of another machine.

Platform - Server Platform

Standard 7.01.04 Use NetWare Directory Services (NDS) for directory services. File and print can use NOS services on Local Area Networks (LANs).

Rationale:

- Directory services are a key component of the enterprise's infrastructure. Standards are being developed for enterprise directory services. NDS is widely available and a useful interim standard affording an easy migration path should it be necessary.
- There is an immediate and continuing demand for the purchase and installation of Local Area Networks (LAN) operating systems or file systems. Although statewide standards for such system components are being developed, they are part of a larger, more complex specifications. In the interim, consistency among immediate purchases, such as NDS, eases the burden of maintenance, may reduce the purchase costs, and will greatly simplify migration to the new standard when it is in place.
- For more information on directory services, see the Groupware Chapter of the Statewide Technical Architecture.

Platform - Server Platform

Best Practice 7.01.04 Balance business adaptability and ease of systems management with server platform choices. However, when there is a conflict between business adaptability and ease of systems management, the business requirements should have highest priority.

Rationale:

- These two goals will always be in conflict.
- The primary design point of the technical architecture is to provide for change in business operations and its supporting applications. Therefore, even though it is easier to manage a large server rather than multiple smaller servers, the business need to provide flexibility should take precedence over any marginal increases in operational costs.

Platform - Server Platform

Standard 7.01.05 Servers must be secured in such a way as to ensure security, availability and reliability.

Rationale:

- Application servers run the state's business. They must be physically secure, reliable and available for processing. In order to ensure this, the following are requirements for servers:
- They must have UPS with a battery backup sufficient to meet the minimum up-time as described by the data criticality.
- Access should be restricted to authorized personnel only.
- Must meet security policy standards.

Platform - Server Platform

Standard 7.01.06 Servers must be secured in such a way as to ensure security, availability and reliability.

Rationale:

- They must have UPS with a battery backup sufficient to meet the minimum up-time as described by the data criticality.
- UPS should be capable of issuing a warning and optionally call via a page the responsible personnel.
- Access should be restricted to authorized personnel only.
- Must meet security policy standards.

Platform - Client Platform

Best Practice 7.02.01 Use open standards based host-controlled client platforms where standards exist.

Rationale:

- Choose a device and host software that is already in use elsewhere in the enterprise. For example, the Deparment of Motor Vehicles is using PDF417 for 2-d bar codes. Unless significant business reasons exist for other choices, use the PDF417 coding stasndard
- Consider other potential uses for the device and host software in the enterprise.

Platform - Client Platform

Standard 7.02.01 Two-dimensional (2-d) bar codes should use PDF417 coding standard.

Rationale:

- The PDF417 bar code standard is used by the Department of Motor Vehicles. It is capable of storing data such as product information, maintenance schedule, shipping information for others.

Platform - Client Platform

Best Practice 7.02.02 Ideally, client platform choices should satisfy both end-user ease-of-use and ease of systems management. When there is a conflict between end-user ease-of-use and ease of systems management, give priority to end-user needs.

Rationale:

- It would simplify systems management to standardize on a particular platform; however, application software is often available only for particular client platforms. Apple, for example, supplies software for education; Unix provides engineering packages, etc. Supporting the business needs of end users is most important.

Platform - Client Platform

Standard 7.02.02 Platforms must comply with third party certifications.

Rationale:

- Client platforms must comply with third party certifications as specified.
  - Unix: Manufacturer is ISO 9002 Certified, XPG4 Branded UNIX 93
  - Microcomputers: Manufacturer is ISO 9002 certified, Gartner Group Tier 1 or Tier 2 classified.

Platform - Client Platform

Standard 7.02.03 ISO 7816 Smart Card standards for contact smart cards.

Rationale:

- See Security and Directory Services Architecture standards for details.

Platform - Client Platform

Best Practice 7.02.03 Choose client platforms that support personal productivity and connectivity. This may require multiple client configurations to support business needs.

Rationale:

- A line of business application may have a variety of users with different client platform requirements. A financial department may use Windows 95 because of the availability of accounting applications while an engineering department may require UNIX because of the availability of CAD/CAM software.
- If personal productivity applications are run on the desktop, a 32-bit or 64-bit operating system is required. Migration from 16-bit operating system platforms to 32- or 64- bit operating system platforms supports faster processing, access to more memory, and better memory and process management. The 32- and 64- bit operating systems provide more stable, reliable platforms in an N-tiered, distributed client/server environment thus reducing the number of system crashes caused by lack of client resources. In addition, 64-bit operating systems provide better performance for process intensive applications such as multi-media and engineering (CAD) applications.

Platform - Client Platform

Best Practice 7.02.04 The client platform displays the interface to an application. In the design of applications, minimize dependency on a particular client platform as much as possible.

Rationale:

- Web browsers support mulitple platforms.
- 3-tier and N-Tier application architectures, using "thin" clients, reduces dependence on a particular client platform because the user interface is isolated from application code (see the Application Arthitecture chapter).

Platform - Client Platform

Standard 7.02.04 ISO 14443A and Mifare Smart Card standards for contactless smart cards.

Rationale:

- See Security and Directory Services Architecture standard for details.

Platform - Client Platform

Standard 7.02.05 Avoid Proprietary smart card reader-side APIs.

Rationale:

- No standards exist for smart card reader-side APIs for application and platform integration. Use reader-side APIs from established platform vendors, such as PC/SC for the windows environment or use APIs that strictly adhere to the ISO 7816/4 command set.
- See Security and Directory Services Architecture standard for more details.

Platform - Client Platform

Standard 7.02.06 Standards for Operating Systems will be covered in a later release of this chapter.

Rationale:

- Operating systems standards will be covered in a later release.

Enterprise Management - Enterprise Management Architecture

Principle 8.00.01 Business needs should have priority when making systems management decisions.

Rationale:

- System management must facilitate the business process. Business unit needs should play a primary role when identifying requirements and selecting technology and applications. Business units are assuming a larger role in driving technology selection and its application.
- Whenever a business need conflicts with a systems management need, the business need must take priority.
- Business units should have as much autonomy as possible to select applications that meet their needs. As long as the business functionality justifies the cost and the business unit is willing to pay the price, then the selected application is acceptable. Support costs should be considered by the business unit.

- To support business processes, systems management must focus on increasing system stability and availability while reducing costs. It can achieve these goals by setting standards, establishing guidelines and centralizing systems management functions along business functional lines.
- Centralization/standardization should occur within a business function. However, a single standard does not apply to all lines of business. For example, all operators using the same system should have a minimum standard hardware and software configuration to meet their needs. Operators using other systems may require different tool sets to meet the needs of their unique business applications. Configurations can be different, however all configurations can be based on the same architectural components.

Enterprise Management - Enterprise Management Architecture

Principle 8.00.02 Organizations should limit the number of permutations in products to facilitate support efforts and reduce long term support costs.

Rationale:

- Uncontrolled product deployment contributes to a level of complexity that exceeds the support capability of current distributed systems management, DSM, tools and increases staff and training costs. Choices for managing this difficult situation include:
- Scaling back deployment to a manageable range.
- Reducing complexity through consistent product selection.
- Planned retirement of applications, hardware and operating systems with performance problems and/or that are difficult or impossible manage and support.
- Deployment of consistent environments enables the systems management infrastructure to adjust to change. For example, when all users of a particular system have a recommended standard desktop software configuration, this common basic environment makes it easier to plan and install system upgrades and to isolate problems.
- A finite and identifiable product universe facilitates centralized support and planned operational changes.
- Careful selection of products that can be supported centrally is more cost effective because it reduces the support burden of `shadow' or peer to peer support. Support costs are one of the most expensive systems management components.
- Established product selection criteria contributes to cost savings through discounts provided for state-wide software product licenses. Business managers need to be aware of the impact that business decisions have on support costs.
- The learning curve and associated training costs for technical staff are reduced when products are carefully selected to comply with architectural requirements.

Enterprise Management - Enterprise Management Architecture

Principle 8.00.03 Limit the amount of "unique" performance tuning to existing individual network components, particularly servers and desktops.

Rationale:

- Performance tuning for unique/non-standard components is not worth the increased maintenance costs of multiple configurations.
- Performance tuning can inhibit change by encouraging comfort with the status quo.

- It may be cheaper to increase performance by upgrading to an architecturally compliant hardware configuration than to spend time tuning an application

Enterprise Management - Enterprise Management Architecture

Principle 8.00.04 Increase capital investment when it offsets long-term support costs.

Rationale:

- Identical configurations are easier to support. In the long term it may be much more expensive to support multiple types of configurations than it is to invest in replacing them with consistent configurations.
- Purchasing hardware that exceeds the immediate need often saves money in the long run, as it promotes expandable systems and reduces the need for tuning and support.
- It is more cost effective to use capital dollars to improve operations than to spend support dollars on outmoded technology. The cost of continuing to support an aged configuration is often higher than the cost of new equipment that will improve performance and reduce support costs.
- The practice of using "hand-me-down" equipment perpetuates obsolete technology and can greatly increase the support burden by increasing the number and kind of devices requiring support and its associated costs.

Enterprise Management - Enterprise Management Architecture

Principle 8.00.05 Utilize open, vendor-neutral standards whenever possible.

Rationale:

- Open, vendor-neutral systems standards provide flexibility and consistency that will allow agencies to respond more quickly to changing business requirements.
- Vendor-neutral systems support economic and implementation flexibility.
- Vendor-neutral systems also protect the state against unexpected changes in vendor strategies and capabilities.

Enterprise Management - Help Desk

Best Practice 8.01.01 The help desk and user support functions must be re-engineered to provide an integrated support services environment.

Rationale:

- The central help desk provides the focal point to mediate problems.
- Support tools should empower both the help desk analyst and the end user with self-help capabilities.

Enterprise Management - Help Desk

Best Practice 8.01.02 The help desk should actively work to improve the perception of its services within the organization.

Rationale:

- Help desk analysts must be empowered to take ownership of problems and given the tools to solve them.
- As part of managing the changing perception of the help desk organization, marketing events, such as newsletters, should target the end-user community as well as their managers.
- Upper management should periodically work on the help desk to demonstrate commitment to service and gain greater appreciation for user needs.
- Training for end users should be included in all help desk improvement plans.

Enterprise Management - Help Desk

Best Practice 8.01.03 In order to provide the best customer service environment, it may be necessary to elevate and/or restructure the help desk within the organization.

Rationale:

- The help desk organization should be elevated in the organizational and reporting structure to operate independently of other units, making customer service needs its top priority.
- The role of the help desk analyst is changing. Help desk staff should serve on project teams, and participate in training, application design, testing, and maintenance.
- All requests for service should be channeled through the help desk when feasible.

Enterprise Management - Help Desk

Best Practice 8.01.04 A single consolidated help desk design supports an enterprise model.

Rationale:

- A consolidated help desk does not have to be physically located in one place. However, it should have one constituency, one phone number, one set of procedures, one set of defined services, and one set of integrated network systems management (NSM) platforms and applications.
- The implementation of the virtual data center (VDC), where many remote LANs are managed as a single entity, supports the corresponding development of consolidated help desk services.

Enterprise Management - Help Desk

Best Practice 8.01.05 Each centralized help desk unit must provide a single point of contact (SPOC).

Rationale:

- A SPOC minimizes user inconvenience and confusion. In its broadest sense, SPOC means that the end user makes one attempt at contact and the help desk request is channeled by some automated means to the organization that can best service the request.
- The help desk should mediate all problems.

Enterprise Management - Help Desk

Best Practice 8.01.06 In order to leverage support resources and provide effective client support, multiple tiers or levels of client support are required.

Rationale:

- Tier/Level 1 client support should have end-to-end responsibility for each client request. The help desk analyst should be empowered to resolve as many requests as possible. Tier 1 provides the client contact point (CCP) or call ownership, which is the single point of contact for the end user to request a service. Organizations should retain control of the Tier 1 help desk in order to ensure the quality of the customer relationship.
- Tier/Level 2 client support provides advanced technical expertise to the tier/level 1 client contact points. Their responsibility is to analyze the requests routed to them and resolve the problems. Resources at this level can be composed of staff specialists and/or third party providers/vendors.
- Tier/Level 3 support is composed of highly specialized technical experts. Calls which cannot be solved at tiers/levels 1 and 2 are routed to this level. Resources at this level can be composed of staff specialists and/or third-party providers/vendors.
Long binary or text value

Enterprise Management - Help Desk

Best Practice 8.01.07 Reliable metrics and reports must be defined and used to assist managers, help desk staff, and the client community to assess the effectiveness of the help desk in meeting organizational goals.

Rationale:

- Both consolidated high level and low level detailed measures are critical to successful service desk operations.
- Metrics should be used to identify trends and to support a proactive management approach that anticipates and avoids problems.
- Monitoring server information and trend analysis of performance statistics for comparing LAN operations generates important information necessary to remotely support many LANs.
- Methods and procedures to solve problems should be developed, published and followed and measured.
- Service level agreements (SLA's) should be developed stating responsibilities of both the help desk and its clients. SLA criteria are one method to evaluate help desk performance.

Enterprise Management - Help Desk

Best Practice 8.01.08 Geographically dispersed help desk units must inter-operate and share information.

Rationale:

- All requests for service should reside in a database that is shared by technology and application-based help desk units serving specific constituencies throughout the state. This process shares information and makes it possible for one help desk to electronically

pass a service request to another help desk without forcing the user to make another contact attempt.
- The use of technological advances, such as distributed processing, dynamic control of users desktop, improved telephony, and client support software, make it possible for geographically dispersed help desk groups to function as a cohesive support unit.

Enterprise Management - Help Desk

Best Practice 8.01.09 Resolution databases that contain solutions to recurring problems should be built to improve service quality and contain costs.

Rationale:

- Building and using a knowledge base of prior resolutions to solve problems improves the quality of resolutions.
- Help desk operations should include problem resolution links to external systems.

Enterprise Management - Help Desk

Best Practice 8.01.10 The help desk should maintain Inventories of hardware and software configurations. They should include all physical components (processor, RAM, disk drive, network cards, add-on cards) and other types of relevant information.

Rationale:

- Current inventories are critical to support functions.
- Inventory "agents" or applications that survey and record current inventory facilitate collection from desktops and servers.

Enterprise Management - Operations Management

Best Practice 8.02.01 Equipment deployed in virtual data centers must be configured to facilitate remote management and support.

Rationale:

- The VDC should be configured to prevent a single point of failure.
- Identical configurations of rack-mounted servers are placed in secure locations (closets).
- For reliability and ease of support, each major application should be placed on a uniformly configured server. This may require that each major application be implemented on its own server. (See the Platform Architecture Chapter.)
- Use the same reference configuration on these servers. Important items to consider when planning for consistency include using the same versions of network software, using the same network hardware cards, etc. (See the Platform Architecture Chapter.)
- Systems management tools, consistently applied, allow management of multiple instances of the identical network configurations at remote sites as if they were on the data center floor.
- The VDC should support mission critical applications.

Enterprise Management - Operations Management

Standard 8.02.01 Use SNMPv1 (simply called SNMP) protocols.

Rationale:

- The Simple Network Management Protocol (SNMP) is a group of internet protocols that is the standard for managing TCP/IP based networks.
- It is built into the devices (e.g., concentrators, routers) in the network and in the network operating systems of the servers and workstations.
- The network management system uses SNMP to collect statistics and other information on network devices.
- SNMP is also used to send commands that control the state of network devices.
- In 1993, SNMPv2 attempted to address security and control issues, however there were inter-operability issues between SNMPv1 and SNMPv2. It is extremely difficult to find an SNMPv2 manager, therefore SNMPv2 has not been accepted and is widely considered a failure.

Enterprise Management - Operations Management

Best Practice 8.02.02 Systems management functions for the virtual data centers should be remotely performed.

Rationale:

Some examples of remote systems management services include:

- Backup, archiving and recovery
- System, database and application monitoring
- Software distribution to the server and/or desktop

Enterprise Management - Operations Management

Standard 8.02.02 Use Remote Monitoring (RMON) products.

Rationale:

- RMON products are predicted to become increasingly used in most enterprise networks.
- RMON products provide packet collection, decoding and analysis to the MAC layer of the Operating Systems Interconnection (OSI) stack using a combination of consoles and hardware and software probes that relied on SNMP MIB data collections.
- In 1992, the Internet Engineering Task Force, IETF, specified the RMON1 standard in RCF 1271. The RMON1 MIB extends SNMP capability by monitoring sub-network operation and reducing the data collection burden on management consoles and network agents.
- The RMON2 standard was approved by the IETF in January 1997 in RCF2021. RMON2 includes a new MIB to extend network monitoring into the application monitoring layer.
- RMON functionality is growing to include functions like applications monitoring, report generation and bandwidth allocation.
- All major network device vendors have added RMON MIB collection capability to their products, although the depth of implementation relative to the full RMON specification varies among vendors and products.

Enterprise Management - Operations Management

Standard 8.02.03 Conform to the Desktop Management Interface (DMI) standard.

Rationale:

- The DMI standard was developed by the Desk Top Management Task Force (DMTF), which sets specifications for the management of the desktop environment.
- The DMI is a set of API's that allow different vendor applications to consistently share the desktop.
- It sets the standard for a management platform that enables a common standardized mechanism for systems management of the desktop while permitting vendor differentiation.
- As vendors build desktops with embedded DMI standards, important desktop management information will become available from the newer desktop units.

Enterprise Management - Operations Management

Best Practice 8.02.03 Under the Virtual Data Center concept, responsibilities of customers for systems management are limited.

Rationale:

Even though the equipment is located close to the customer community, for the most part, local user efforts should be concentrated on performing their business functions rather than on system management tasks such as system configuration, debugging and/or backup.

Enterprise Management - Operations Management

Best Practice 8.02.04 System components should proactively alert in advance of failure including predictive capability.

Rationale:

System generated alarms and alerts should be automatically routed to the appropriate systems management resource. For example:

- Database problems should be routed to the database support group.
- PC hardware problems should be routed to PC support.
- Agents should be able to issue alerts for both hardware and applications.

Enterprise Management - Operations Management

Best Practice 8.02.05 Inventories of hardware and software configurations should be maintained real-time.

Rationale:

- Inventories of configurations are critical to support functions
- Inventory capability requires `agents' on workstations and servers.